Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Лесосибирский педагогический институт –
филиал Сибирского федерального университета

Кафедра высшей математики, информатики и естествознания

УТВЕРЖДАЮ
Заведующий кафедрой
_____ Н.Ф. Романцова
подпись
« 8 » июня 2018 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.02 Информационные системы и технологии

Разработка автоматизированной информационной системы «Автопарк такси Бонус»

Руководитель _____ 04.06.18 доцент, канд.пед.наук С.С.Ахтамова
подпись, дата

Руководитель _____ 04.06.18 доцент, канд.экон.наук А.В.Рубцов
подпись, дата

Выпускник _____ 7.06.18 И.А.Якушкин
подпись, дата

Лесосибирск 2018

РЕФЕРАТ

Выпускная квалификационная работа содержит 70 страниц текстового документа, 22 рисунка, 5 таблиц, 20 использованных источников, 2 приложения.

АВТОМАТИЗАЦИЯ, РАЗРАБОТКА, ПРОЕКТИРОВАНИЕ, ТАКСИ.

Целью выпускной квалификационной работы является проектирование автоматизированной информационной системы автопарка такси.

Объект исследования – предприятие ООО «Такси Бонус».

Предмет исследования – проектирование автоматизированной информационной системы автопарка такси.

Основные задачи:

– рассмотреть деятельность предприятия в области учета автопарка такси и его работы;

– обосновать потребность в автоматизации;

– выполнить обзор и выбор инструментальных средств проектирования и реализации системы;

– разработать и выполнить поэтапное тестирование автоматизированной информационной системы.

В данной работе описывается процесс проектирования и тестирования информационной системы, которая служит для автоматизации процесса учета перевозок службой такси.

В качестве СУБД выбрана MySQL, в качестве языка программирования Delphi, в качестве среды разработки выбрана Delphi 2010.

В работе показан поэтапный процесс проектирования программного обеспечения, начиная от постановки задачи, разработки инфологической модели данных до проектирования алгоритмов системы и их реализация на языке программирования.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

На современном уровне развития автоматизация процессов представляет собой один из подходов к управлению процессами на основе применения информационных технологий. Этот подход позволяет осуществлять управление операциями, данными, информацией и ресурсами за счет использования компьютерной техники и программного обеспечения, которые сокращают степень участия человека в процессе, либо полностью его исключают [2].

Основной целью автоматизации является повышение качества бизнес-процесса. Автоматизированный процесс обладает более стабильными характеристиками, чем процесс, выполняемый в ручном режиме. Во многих случаях автоматизация процессов позволяет повысить производительность, сократить время выполнения процесса, снизить стоимость, увеличить точность и стабильность выполняемых операций [6].

Актуальность данной работы заключается в том, что исследуемая организация для ведения документооборота, сопровождающего основную деятельность, использует бумажные журналы учета, которые не позволяют вести оперативный учет работы автопарка и затрудняют доступ к информации, существенно замедляет ее анализ.

Целью выпускной работы является проектирование автоматизированной информационной системы для предприятия.

Объектом исследования является предприятие ООО «Такси Бонус».

Предметом исследования является проектирование автоматизированной информационной системы автопарка такси.

Для достижения поставленной цели требуется решить такие задачи:

- рассмотреть деятельность предприятия в области учета автопарка такси и его работы;
- обосновать потребность в автоматизации;

– выполнить обзор и выбор инструментальных средств проектирования и реализации системы;
– разработать и выполнить поэтапное тестирование автоматизированной информационной системы.

1 Анализ предметной области

1.1 Характеристика предприятия и его деятельности

В работе исследуется деятельность предприятия ООО «Такси Бонус».

«Такси Бонус» – это активно развивающееся молодое предприятие, предоставляющее услуги такси в городе Лесосибирске. Компания была основана в феврале 2014 года. За это время предприятию удалось не только удержаться, но и прочно закрепиться на современном конкурентном рынке услуг.

Мировая история такси сегодня насчитывает более 370 лет. Именно тогда, в далеком 1636 году, был придуман не только новый вид общественного транспорта, но и разработан целый свод негласных правил, повышающих качественный уровень перевозок пассажиров. Основываясь на этом, на начальном этапе становления компании, был разработан ряд принципов, которых и сегодня придерживается каждый сотрудник в своей повседневной практике общения с клиентами:

– пунктуальность;

– комфорт;

– вежливость;

– экономичность.

Ежедневно такси принимает десятки вызовов из самых разных уголков Красноярского края. Но, несмотря ни на что, такси старается выполнять свои обязательства в срок. Время клиента – это основная ценность в работе компании.

Высокий качественный уровень услуг такси, которого достигла компания за незначительный срок своей работы – это умение рассчитывать свое время, с учетом загруженности дорог, возможных климатических изменений таким

образом, чтобы каждая машина прибывала на вызов в точно установленное время.

Постепенное пополнение автопарка позволяло с каждым днем расширять географию вызовов. Сегодня такси активно предоставляет свои услуги в двух городах и по Красноярскому краю. Расширяющийся автопарк, который в настоящее время насчитывает более десятка различных машин, позволяет не только оперативно реагировать на каждый вызов, но и удовлетворить любые требования клиента (наличие кресла-люльки, детского кресла, курящего салона, перевозка крупного багажа и многое другое).

Лояльная ценовая политика – еще одна неотъемлемая черта организации. Индивидуальный подход к каждому клиенту – это залог успешного сотрудничества предприятия и клиента.

Организационная структура такси представлена на рисунке 1.

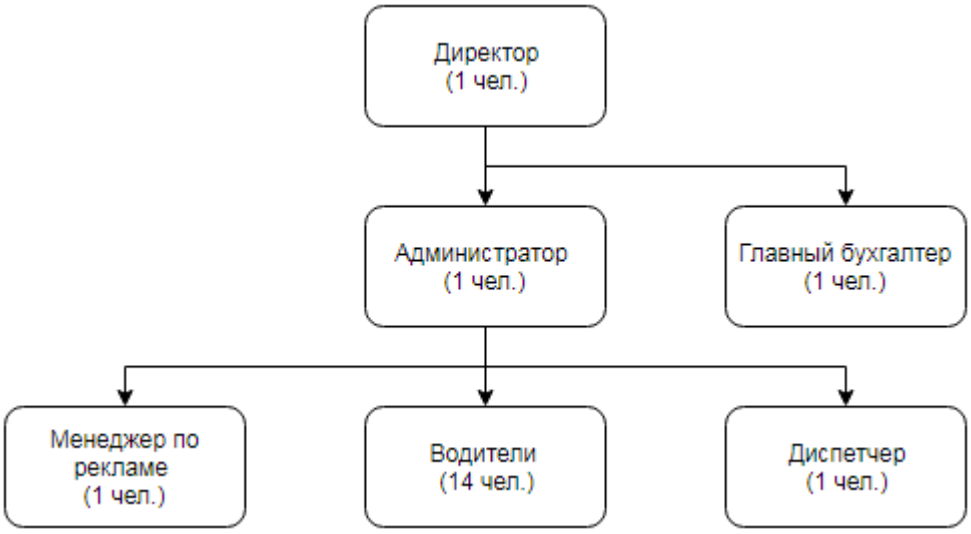


Рисунок 1 – Организационная структура такси

Как видно из рисунка, организационная структура исследуемого предприятия достаточно проста и является линейной. Такая структура свойственна молодым и небольшим предприятиям и предполагает четкую

иерархию и подчиненность нижестоящих уровней по отношению к вышестоящим [15].

В линейной структуре система управления предприятием компонуется по производственному признаку с учетом особенностей деятельности [16].

Численность персонала организации на период исследования составляет 19 человек.

1.2 Теоретический обзор и выбор методологии и средств проектирования автоматизированной системы

Следует отметить, что существует два базовых подхода к проектированию автоматизированных информационных систем: структурный и объектно-ориентированный.

Сущность структурного подхода при разработке ИС заключается в ее декомпозиции на автоматизируемые функции. При этом систему разбивают на функциональные подсистемы. Они, в свою очередь, делятся на подфункции, которые подразделяются на задачи и так далее. Этот процесс разбиения продолжается до конкретных процедур [8].

Отметим, что автоматизируемая система при этом сохраняет целостное представление, которое связывает все ее компоненты. При разработке системы по методологии «снизу-вверх» от отдельных подзадач до всей системы целостность теряется, и возникают проблемы при проектировании информационной стыковки некоторых компонентов.

Использование же объектно-ориентированных методов позволяет создать модель (описание) предметной области как совокупность объектов – сущностей, которые объединяют данные и методы их обработки (процедуры).

Каждый такой объект обладает собственным поведением и моделирует определенный объект реального мира. С данной точки зрения объект представляет вполне осязаемую вещь, демонстрируя определенное поведение.

В данном подходе фокус внимания переносится на характеристики физической (иногда абстрактной) системы, которая является предметом моделирования. Объекты при этом обладают целостностью, и она не может быть нарушена. То есть свойства, которые характеризуют объект и его поведение, остаются постоянными. Объект может менять состояние или становиться в отношение к другим объектами.

Классификация методов проектирования [14]:

а) по степени автоматизации разработки проектных решений:

– ручного проектирования, метод, при котором проектирование компонентов АИС выполняется без использования специальных программных инструментов, а программирование выполняется на алгоритмических языках;

– компьютерного проектирования, метод, который выполняет генерацию или конфигурацию (настройку) решений для проекта на основе использования специальных программных инструментов.

б) по степени типизации проектных решений:

– оригинального (индивидуального) проектирования, метод, при котором проектные решения разрабатывают «с нуля» в соответствии с требованиями, предъявленными заказчиком к АИС, метод характеризуется тем, что все виды проектных работ ориентированы на создание индивидуальных для каждого объекта ИС проектов, которые максимально отражают все его особенности;

– типового проектирования, метод, который предполагает конфигурацию АИС из готовых типовых проектных решений (программных модулей), выполняется на основе опыта, полученного при разработке индивидуальных проектов.

в) по степени адаптивности проектных решений:

– реконструкции, метод, в котором адаптация проектных решений выполняют путем переработки соответствующих компонентов АИС (перепрограммирования программных модулей);

– параметризации, когда проектные решения настраиваются в соответствии с изменяемыми параметрами;

– реструктуризации модели, когда изменяется модель проблемной области, на основе которой автоматически модифицируются проектные решения.

Проектирование информационных систем предполагает использование различных средств проектирования:

– нормативных и правовых документов;

– систем классификации и кодирования информации;

– систем проектной документации;

– моделей информационной системы и их составных частей;

– методик анализа и принятия проектных решений;

– программных средств.

Иерархия технологий проектирования:

1) Каноническое проектирование – проектирование ИС направленное на отражение особенностей технологии индивидуального (оригинального) проектирования. Среди основных характерных особенностей канонического проектирования можно выделить такие особенности, как:

– отражение особенностей ручной технологии проектирования;

– ориентация на индивидуальное (оригинальное) проектирование;

– осуществление на уровне исполнителей;

– возможность интеграции выполнения элементарных операций; применение, как правило, для сравнительно небольших, локальных ИС;

– использование инструментальных средств универсальной компьютерной поддержки.

Каноническое проектирование направлено на минимальное использование типовых проектных решений. Адаптация проектных решений

при каноническом проектировании осуществляется только путем перепрограммирования соответствующих программных модулей.

2) Индустриальное проектирование – проектирование, ориентированное на использование современных средств автоматизации. Индустриальное проектирование в последнее время получило широкое применение, однако это не исключает использования в отдельных случаях канонической технологии. К основным особенностям индустриального проектирования относятся:

– используется для проектирования ИС, автоматизирующей работу с большими и сложными объектами;

– предполагает широкое применение интегрированных средств автоматизации проектирования;

– может использоваться как при выполнении оригинальных, так и типовых проектов;

– обычно сочетается с использованием итерационной модели ЖЦ ИС;

– обычно связано со значительной реорганизацией процессов производства и/или управления ОА.

Для реализации типового проектирования используются два подхода: параметрически-ориентированное и модельно-ориентированное проектирование.

Параметрически-ориентированное проектирование включает в себя следующие этапы: определение критериев оценки пригодности пакетов прикладных программ (ППП) для решения поставленных задач, анализ и оценка доступных ППП по сформулированным критериям, выбор и закупка наиболее подходящего пакета, настройка параметров (доработка) закупленного ППП.

Критерии оценки ППП делятся на следующие группы:

– назначение и возможности пакета;

– отличительные признаки и свойства пакета;

– требования к техническим и программным средствам;

– документация пакета;

– факторы финансового порядка;

– особенности установки пакета;

– особенности эксплуатации пакета;

– помощь поставщика по внедрению и поддержанию пакета;

– оценка качества пакета и опыт его использования;

– перспективы развития пакета.

Внутри каждой группы критериев выделяется некоторое подмножество частных показателей, детализирующих каждый из десяти выделенных аспектов анализа выбираемых ППП.

Числовые значения показателей для конкретных ППП устанавливаются экспертами по выбранной шкале оценок (например, 10-бальной). На их основе формируются групповые оценки и комплексная оценка пакета (путем вычисления средневзвешенных значений). Нормированные взвешивающие коэффициенты также получаются экспертным путем.

Модельно-ориентированное проектирование заключается в адаптации состава и характеристик типовой ИС в соответствии с моделью объекта автоматизации.

Технология проектирования в этом случае должна обеспечивать единые средства для работы, как с моделью типовой ИС, так и с моделью конкретного предприятия.

Типовая ИС в специальной базе метаинформации – репозитории – содержит модель объекта автоматизации, на основе которой осуществляется конфигурирование программного обеспечения.

Таким образом, модельно-ориентированное проектирование ИС предполагает, прежде всего, построение модели объекта автоматизации с использованием специального программного инструментария (например, SAP Business Engineering Workbench (BEW), BAAN Enterprise Modeler). Возможно также создание системы на базе типовой модели ИС из репозитория, который

поставляется вместе с программным продуктом и расширяется по мере накопления опыта проектирования информационных систем для различных отраслей и типов производства.

Репозиторий должен содержать базовую (ссылочную) модель ИС, типовые (референтные) модели определенных классов ИС, модели конкретных ИС предприятий.

Основными современными методологиями при проектировании информационных систем являются [1]:

– методология функционального моделирования работ SADT;

– методология RAD – быстрой разработки приложений;

– методология Rational Unified Process (RUP).

Методология SADT представляет собой методологию структурного анализа и проектирования и базируется на структурном анализе различных систем и графическом представление исследуемого предприятия в виде системы функций, имеющей три класса моделей:

– функциональная модель;

– динамическая модель;

– информационная модель.

Процесс моделирования согласно методологии SADT включает такие этапы:

– сбор информации, а также анализ данных о предметной области;

– документирование собранной информации;

– моделирование (например, в нотации IDEF0);

– коррекция модели.

Принципы методологии RAD были реализована в кратчайшие сроки группой разработчиков с использованием так называемого «инкрементного прототипирования». Это позволило на ранней стадии проектирования системы

продемонстрировать действующую интерактивную модель прототипа системы, а также уточнить принятые решения и оценить некоторые характеристики.

Методология RAD включает такие стадии:

– моделирование потоков информации между функциями;

– моделирование данных;

– изменение объектов данных, которые обеспечивают реализацию бизнес-функций;

– создание приложений;

– тестирование.

Недостатками методологии RAD являются:

– требуется большой коллектив разработчиков для больших информационных систем;

– не используется при применении новых технологий;

– применяется для таких информационных систем, которые могут быть разделены на отдельные модули, а также в которых не сильно высокие требования к производительности.

Методология RUP реализует подходы:

– итерационный и инкрементный;

– построение системы на основе архитектуры информационной системы;

– управление проектом на базе функциональных требований к системе.

Разработка системы выполняется итеративно. Включает в себя отдельные проекты, небольшие по объему, включающие свои собственные этапы: составления требований, проектирования, а также реализации, тестирования и интеграции. Итерации заканчиваются созданием работающей подсистемы.

Итерационный цикл можно охарактеризовать периодической обратной связью возможность адаптации к ядру разрабатываемой системы. Таким образом, создаваемая система растет и совершенствуется постепенно.

CASE-средства представляют собой специальный набор применяемой техники, а также методов программной инженерии при создании программного продукта, помогающие обеспечить отсутствие ошибок, высокое качество, а также простое обслуживание программного продукта [4].

Главной целью CASE-средств является увеличение производительности труда разработки, а также облегчение работы разработчиков программного продукта.

CASE-средство состоит прежде всего из:

– методологии – задает единый графический язык, а также правила и методы работы с ним;

– графических редакторов – используются при построении диаграмм;

– генератора – генерирует исходный код для различных платформ;

– репозитория – база данных, хранящая результаты работы разработчиков.

Под классификацией понимают подчинение нескольких предметов, явлений по некоторым признакам. Все CASE-средства можно распределить по нескольким классификациям.

Так, по функциональной направленности CASE-средства можно разделить на:

– средства анализа и проектирования;

– средства проектирования баз данных и файлов;

– средства программирования;

– средства сопровождения и реинжиниринга;

– средства окружения;

– средства управления проектом.

В качестве средства анализа и проектирования предлагается использовать BPWin, а также ERWin.

Возможности BPwin [10]:

– поддерживает сразу три стандартные нотации – IDEF0 (функциональное моделирование), DFD (моделирование потоков данных) и IDEF3 (моделирование потоков работ). Эти три основных ракурса позволяют описывать предметную область наиболее комплексно;

– позволяет оптимизировать процедуры в компании;

– полностью поддерживает методы расчета себестоимости по объему хозяйственной деятельности (функционально-стоимостной анализ, ABC);

– позволяет облегчить сертификацию на соответствие стандартам качества ISO9000;

– интегрирован с ERwin (для моделирования БД), Paradigm Plus (для моделирования компонентов ПО) и др.;

– интегрирован со средством имитационного моделирования Arena;

– содержит собственный генератор отчетов;

– позволяет эффективно манипулировать моделями – сливать и расщеплять их;

– имеет широкий набор средств документирования моделей, проектов.

Пакет ERWin это средство концептуального моделирования БД. Используется при моделировании и создании баз данных произвольной сложности на основе диаграмм "сущность – связь". В настоящее время ERWin является наиболее популярным пакетом моделирования данных благодаря поддержке широкого спектра СУБД самых различных классов. Возможности ERWin [19]:

– поддерживает методологию структурного моделирования SADT и следующие нотации: стандартную нотацию IDEF1x для ER-диаграмм моделей данных, нотацию IE и специальную нотацию, предназначенную для проектирования хранилищ данных – Dimensional;

– поддерживается прямое (создание БД на основе модели) и обратное (генерация модели по имеющейся базе данных) проектирование для 20 типов

СУБД: настольные, реляционные и специализированные СУБД, предназначенные для создания хранилищ данных;

– интегрирован линейкой продуктов Computer Associates для поддержки всех стадий разработки ИС, CASE-средствами Oracle Designer, Rational Rose, средствами разработки и др.;

– позволяет повторно использовать компоненты созданных ранее моделей, а также использовать наработки других разработчиков;

– возможна совместная работа группы проектировщиков с одними и теми же моделями (с помощью AllFusion Model Manager);

– позволяет переносить структуру БД (не сами данные!) из СУБД одного типа в СУБД другой;

– позволяет документировать структуру БД.

1.3 Моделирование деятельности предприятия

Итак, в рамках исследования бизнес процессов предприятия выбран стандарт IDEF0, поскольку он позволяет четко структурировать последовательность операций и определяет участников процесса, входные и выходные параметры, а также окружение выполнения процесса. В качестве средства моделирования будет использован BPWin.

Методология SADT и ее развитие – стандарты IDEF – широко используются в мире самыми разными компаниями в самых разных областях деятельности [7].

Самый распространенный стандарт IDEF0 представляет собой метод структурно-функционального моделирования, описывающий бизнес-процессы в виде иерархической системы взаимосвязанных функций (рисунок 2).

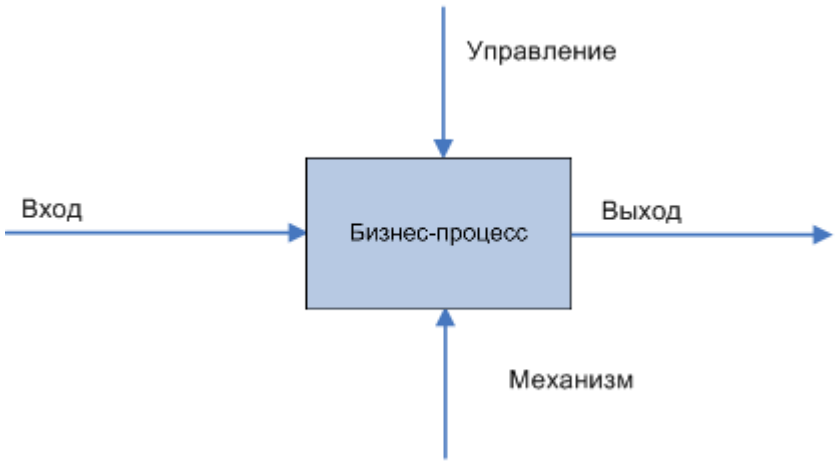


Рисунок 2 – Спецификация сторон блоков в стандарте IDEF0

Смысл схемы, представленной на рисунке выше, заключается в следующем: в результате выполнения процесса, «вход» под воздействием «управления» преобразуется в «выход» посредством «механизма» (исполнителя).

Выходы одного блока могут быть входами, или управлениями, или механизмами (исполнителями) для других. Взаимосвязи блоков друг с другом и с внешней средой отражаются интерфейсными дугами. Интерфейсные дуги изображаются стрелками, ориентация которых отображает направление потоков объектов (материальных, финансовых, информационных).

Перечислим этапы построения системы бизнес-процессов в стандарте (нотации) IDEF0 [13]:

а) Построение модели процесса в нотации IDEF0 начинается с построения общего описания процесса и его взаимодействия с окружающим миром. Эта модель представляется в виде черного ящика и называется диаграммой нулевого уровня (или контекстной диаграммой).

На контекстной диаграмме должны быть отражены [5]:

– цель моделирования (например: разработка нового процесса, сокращение времени выполнения процесса и т.д.);

– точка зрения (директор организации, IT-специалист, финансовый специалист и т.д.).

б) Диаграммы следующих уровней детализируют вышестоящий процесс до уровня, на котором получают ответы на поставленные вопросы. Число уровней декомпозиции ничем не ограничивается, кроме целесообразности и здравого смысла.

Довольно часто на диаграммах верхних уровней не показываются механизмы (ресурсы), если это не является действительно нужной информацией. Так поступают для того, чтобы не «перегружать» модель.

в) На каждом уровне декомпозиции рекомендуется размещать от 2-х до 7 блоков.

г) Все объекты диаграммы должны иметь текстовые описания.

д) Построение декомпозиции – процесс многовариантный и творческий.

Таким образом, произведем моделирование основных бизнес процессов предприятия AS IS в области управления кадрами с использованием программного приложения BPWin 4.1 и технологии IDEF0.

Представим модель AS IS интересующих бизнес процессов. Составим модель AS-IS (рисунок 3).

Рисунок 3 – Контекстная диаграмма БП «Учет деятельности такси «Бонус»

На вход бизнес процесса поступают заявки клиента, а на выходе получается журнал вызовов такси.

Бизнес процесс функционирует в рамках действующего законодательства.

В выполнении бизнес процесса принимает участие диспетчер такси.

Декомпозиция диаграммы представлена на рисунке 4.

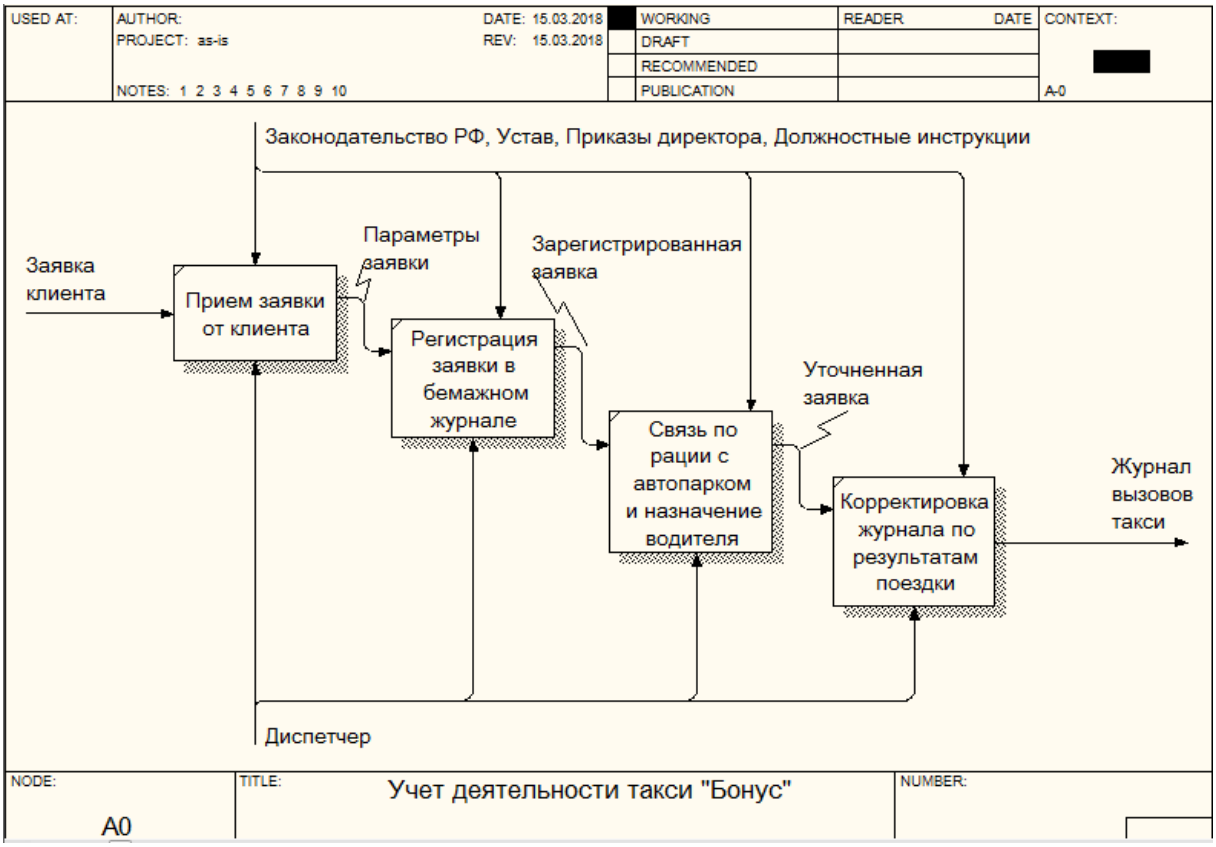


Рисунок 4 – Декомпозиция контекстной диаграммы

В результате декомпозиции получилось четыре подпроцесса:

а) прием заявки от клиента;

б) регистрация заявки в бумажном журнале;

в) связь по рации с автопарком и назначение водителя;

г) корректировка журнала по результатам поездки.

Недостатком существующего учета деятельности такси является то, что применяется бумажный журнал учета, которые не позволяет вести оперативный учет и доступ к информации, серьезно затрудняет и замедляет ее анализ.

Поэтому необходимо перевести учет в электронный вид, причем организовать это стоит при помощи информационной системы, чтобы в дальнейшем была возможность расширять функционал по мере роста компании.

1.4 Анализ различных способов приобретения АИС

Типовыми решениями при проектировании АИС будут являться:

– покупка готовой системы;

– реализация АИС самостоятельно;

– реализация АИС сторонним разработчиком;

– покупка АИС для ее последующей доработки.

Сравним данные способы и выберем подходящий (таблица 1).

Таблица 1 – Сравнение способов приобретения АИС

Способ приобретения \ критерии	Покупка готовой системы	Реализация АИС самостоятельно	Реализация АИС сторонним разработчиком	Покупка АИС для ее доработки
Соответствие ПО поставленной задаче	Частичное	Полное	Частичное	Полное
Адаптивность и масштабируемость	Слабая	Сильная	Средняя	Средняя
Надежность ИС	Фирма-разработчик гарантирует надежность	Потенциально недостаточная надежность	Фирма-разработчик гарантирует надежность	Фирма-разработчик гарантирует надежность

Покупка полностью готового коробочного решения не подходит, потому что вследствие особенностей работы предприятия сложно найти программный продукт, который бы с нуля подходил под данное предприятие. Потребуется его адаптация, а это затраты как на покупку, так и на доработку. К тому же по мере возникновения новых требований придется постоянно обращаться за доработками к компании-разработчику или ее партнерам, что потребует как времени так и дополнительных финансовых затрат.

Покупка и доработка коробочного решения не подходит в силу описанных выше аргументов для покупки готовой системы, ведь покупка готовой системы все равно сведется к ее доработке под задачи предприятия.

Наем компании для разработки ИС позволит реализовать именно тот функционал, который требуется на текущий момент, однако за это придется заплатить большие деньги, и при возникновении новых доработок потребуется опять платить компании. Поскольку разработка систем с нуля у компаний стоит дорого, причем, чем более крупная и известная компания, тем, зачастую, более дорого у нее стоит заказать разработку приложения. Поэтому заказывать придется у небольшой компании-разработчика. Но в таком случае нет никакой гарантии, что через некоторое время эта компания не прекратит свое существование в силу объективных причин, и более нельзя будет производить доработку купленного у нее приложения.

Для того чтобы обеспечить простоту функционирования, гибкость настройки, полную совместимость с требованиями, масштабируемость, следует остановить выбор на самостоятельной реализации АИС.

Следует перейти к непосредственному проектированию системы.

Вывод: в рамках первого раздела выполнен анализ предметной области. Рассмотрено предприятие ООО «Такси Бонус». Выполнен теоретический обзор методологий проектирования. В качестве средства анализа и проектирования предложено использовать BPWin, а также ERWin. Выполнено моделирование существующего бизнес процесса «Учет деятельности такси «Бонус» в нотации IDEF0. Недостатком существующего учета деятельности такси является то, что применяется бумажный журнал учета, которые не позволяет вести оперативный учет и доступ к информации, серьезно затрудняет и замедляет ее анализ. Поэтому предложено перевести учет в электронный вид, причем организовать это стоит при помощи информационной системы, чтобы в дальнейшем была возможность расширять функционал по мере роста компании. В качестве способа реализации АИС выбрана самостоятельная разработка.

2 Проектирование и тестирование АИС «Автопарк такси «бонус»

2.1 Постановка задачи на проектирование

На основе собранной информации, используя ГОСТ 34.602.89, составляем техническое задание на разработку системы.

Техническое задание на систему информационная система «АИС-Такси».

1. Общие сведения.

1.1. Наименование и область применения

Наименование: Информационная система «АИС-Такси».

1.2. Область применения

Предназначена для автоматизации бизнес процессов учета и обслуживания заявок на перевозки пассажиров.

2. Цели разработки.

Целью разработки является повышение эффективности функционирования службы такси.

3. Характеристика объекта автоматизации.

Недостатками существующего учета деятельности такси является то, что применяется бумажный журнал учета, который не позволяет вести оперативный учет и доступ к информации, серьезно затрудняет и замедляет ее анализ.

Эти недостатки обуславливают потребность организации в разработке и внедрении «АИС-Такси».

4. Общие требования к системе.

4.1. Состав и структура ИС

ИС «АИС-Такси» должна быть реализована в виде АРМ, доступ к которому может осуществляться с рабочих станций, подключенных к локальной вычислительной сети предприятия, а также из сети интернет.

Архитектура ИС «АИС-Такси» должна иметь двухуровневую структуру.

Для сохранности информации при сбоях системы и постоянном пользовании должно быть предусмотрено резервное копирование ИС «АИС-Такси» в архив либо средствами самой ИС, либо СУБД.

4.2. Функциональные характеристики ИС

Автоматизированная информационная система должна:

а) позволять вносить в базу новые перевозки;

б) редактировать вспомогательные справочники;

в) позволять выгрузку данных по перевозкам в Excel формат;

г) обеспечивать защиту информации механизмом авторизации и разграничением прав доступа.

Пользователями системы будут являться:

– Администратор, который имеет полный доступ к редактированию таблиц, в том числе таблиц пользователей и назначение им прав;

– Диспетчер, который создает и изменяет заявки на перевозки.

Следует отметить, что каждая заявка при регистрации приобретает свой уникальный номер.

Все переданные в базу данных ИС заявки автоматически отображаются в системе.

4.3. Технические требования к системе.

4.3.1. Требования к программному обеспечению.

Программное обеспечение (ПО) АРМ ИС должно быть написано на стандартных языках программирования, и должно работать в стандартных операционных системах Windows 7/8/10.

ПО должно:

– обладать функциональной полнотой для выполнения всех функций системы (п.4.2. настоящего ТЗ);

– обладать модульностью построения;

– иметь стандартизованные информационные связи и использовать стандартные интерфейсы;

– допускать расширение функциональных возможностей системы. ИС строится таким образом, чтобы легко адаптироваться к изменяющимся условиям, в которых приходится работать организации;

– система должна быть открытой и распределённой;

– администрирование выполняется только в одном месте;

– обновление программного обеспечения должно производиться однократно на сервере, а не на каждой рабочей станции;

– программное обеспечение не должно быть чувствительно к программной и аппаратной платформе;

– наглядный, интуитивный интерфейс.

Качество информационной системы главным образом зависит от качества и достоверности данных, которые в ней хранятся. Поэтому необходимо уделять большое внимание к технологиям сбора данных.

4.3.2. Требования к техническому обеспечению системы.

Требования к пользовательским компьютерам.

Для нормальной работы с ИС «АИС-Такси» рабочие места пользователей должен иметь характеристики не ниже указанных:

– операционная система Windows 7/8/10;

– ОЗУ 2 Гб и выше;

– экран разрешением не менее 1280x800;

– сетевая карта;

– принтер формата А4.

4.3.3. Требования к организационному обеспечению.

ИС «АИС-Такси» эксплуатируется диспетчером такси, а также руководством предприятия с целью контроля деятельности службы.

2.2 Выбор языка программирования и среды разработки

Представим характеристику популярных языков программирования и сред разработки для них [17, 20].

IDE Delphi – язык визуального программирования, предоставляющий разработчику большой объем возможностей по созданию программ, предназначенных для работы с базами данных. Delphi предоставляет разработчику приложения широкие возможности быстрого и качественного проектирования графического интерфейса пользователя – различных окон, кнопок, меню и т.д.

C++ Builder – это продукт фирмы Borland, предназначенный для быстрой разработки приложений (RAD – rapid application development) на языке Си++. С помощью C++Builder можно создавать как консольные приложения Win32, так и использовать графический интерфейс пользователя (GUI – graphical user interface).

MS Visual Studio – это решение для разработки, позволяющее командам любого размера проектировать и создавать привлекательные приложения, которые удовлетворят самым взыскательным требованиям заказчиков. MS Visual Studio включает средства моделирования, обнаружения и проектирования архитектуры, что позволяет описать создаваемую систему и полностью реализовать концепцию программного продукта. В MS Visual Studio присутствуют механизмы, позволяющие получать подробную информацию о возникающих ошибках.

Осуществим сравнение описанных выше языков программирования. Сравнение языков программирования проводилось самостоятельно на базе заявленных производителем технических возможностей языков программирования с использованием метода сравнения.

Результаты сравнительной оценки (по 5 бальной шкале) рассматриваемых средств разработки программного продукта приведены в таблице 3. Данное

сравнение выполнено эмпирически на основе экспертных оценок автора работы.

Таблица 3 – Эмпирическое сравнение средств разработки

Параметр сравнения	IDE Delphi (язык Delphi)	Microsoft Visual Studio (язык C#)	C++ Builder (язык C++)
Быстрота создания программного продукта (знание продукта разработчиком)	4	5	4
Наличие и доступность документации	5	5	5
Поддержка со стороны производителя	5	5	5
Инструментарий средств разработки	5	5	5
Ориентированность на разработку ИС	5	5	3
Итого:	24	5	22

Из таблицы видно, что наиболее оптимальным языком программирования и одноименной (с 2009 года) средой разработки является Delphi. Эти средства позволяют создать полноценное клиентское приложение, которое тесно связанно с базами данных.

2.3 Выбор СУБД

Теперь перейдем к сравнению и выбору СУБД. Выполним сравнение бесплатных и популярных СУБД:

а) MySql [11];

б) Postgresql [18].

Ниже представлены критерии и комментарии по данным критериям.

Хранение данных

MySQL – это реляционная база данных, для хранения данных в таблицах используются различные движки, но работа с движками спрятана в самой системе. На синтаксис запросов и их выполнение движок не влияет. Поддерживаются такие основные движки MyISAM, InnoDB, MEMORY,

Berkeley DB. Они отличаются между собой способом записи данных на диск, а также методами считывания.

Postgresql представляет из себя объектно реляционную базу данных, которая работает только на одном движке – storage engine. Все таблицы представлены в виде объектов, они могут наследоваться, а все действия с таблицами выполняются с помощью объективно ориентированных функций. Как и в MySQL все данные хранятся на диске, в специально отсортированных файлах, но структура этих файлов и записей в них очень сильно отличается.

Стандарт SQL

Независимо от используемой системы управления базами данных, SQL – это стандартизированный язык выполнения запросов. И он поддерживается всеми решениями, даже MySQL или Postgresql. Стандарт SQL был разработан в 1986 году и за это время уже вышло нескольких версий.

MySQL поддерживает далеко не все новые возможности стандарта SQL. Разработчики выбрали именно этот путь развития, чтобы сохранить MySQL простым для использования. Компания пытается соответствовать стандартам, но не в ущерб простоте. Если какая-то возможность может улучшить удобство, то разработчики могут реализовать ее в виде своего расширения не обращая внимания на стандарт.

Postgresql – это проект с открытым исходным кодом, он разрабатывается командой энтузиастов, и разработчики пытаются максимально соответствовать стандарту SQL и реализуют все самые новые стандарты. Но все это приводит к ущербу простоты. Postgresql очень сложный и из-за этого он не настолько популярен как MySQL.

Возможности обработки

Из предыдущего пункта выплывают и другие отличия postgresql от mysql, это возможности обработки данных и ограничения. Естественно, соответствие более новым стандартам дает более новые возможности.

При выполнении запроса MySQL загружает весь ответ сервера в память клиента, при больших объемах данных это может быть не совсем удобно. В основном по функциям Postgresql превосходит Mysql, дальше рассмотрим в каких именно.

Postgresql поддерживает использование курсоров для перемещения по полученным данным. Вы получаете только указатель, весь ответ хранится в памяти сервера баз данных. Этот указатель можно сохранять между сеансами. Здесь поддерживается построение индексов сразу для нескольких столбцов таблицы. Кроме того, индексы могут быть различных типов, кроме hash и b-tree доступны GiST и SP-GiST для работы с городами, GIN для поиска по тексту, BRIN и Bloom. Postgresql поддерживает регулярные выражения в запросах, рекурсивных запросов и наследования таблиц. Но тут есть несколько ограничений, например, вы можете добавить новое поле только в конец таблицы.

Производительность

Базы данных должны обязательно быть оптимизированы для окружения, в котором вы будете работать. Исторически так сложилось, что MySQL ориентировалась на максимальную производительность, а Postgresql разрабатывалась как база данных с большим количеством настроек и максимально соответствующую стандарту. Но со временем Postgresql получил много улучшений и оптимизаций.

В большинстве случаев для организации работы с базой данных в MySQL используется таблица InnoDB, эта таблица представляет из себя B-дерево с индексами. Индексы позволяют очень быстро получить данные из диска, и для этого будет нужно меньше дисковых операций. Но сканирование дерева требует нахождения двух индексов, а это уже медленно.

Все это значит, что MySQL будет быстрее Postgresql только при использовании первичного ключа.

Вся заголовочная информация таблиц Postgresql находится в оперативной памяти. Вы не можете создать таблицу, которая будет не в памяти. Записи таблицы сортируются по индексу, а поэтому вы можете их очень быстро извлечь. Для большего удобства вы можете применять несколько индексов к одной таблице.

В целом PostgreSQL работает быстрее, за исключениям использования первичных ключей. Давайте рассмотрим несколько тестов с различными операциями:

Типы данных

Один из основных моментов обоих баз данных это поддерживаемые типы данных, которые вы можете использовать. Поскольку оба решения пытаются соответствовать синтаксису SQL, то они имеют похожие наборы, но все же кое-чем отличаются.

MySQL поддерживает такие типы данных:

– TINYINT: очень маленькое целое;

– SMALLINT: маленькое целое;

– MEDIUMINT: целое среднего размера;

– INT: целое нормального размера;

– BIGINT: большое целое;

– FLOAT: знаковое число с плавающей запятой одинарной точности;

– DOUBLE, DOUBLE PRECISION, REAL: знаковое число с плавающей запятой двойной точности;

– DECIMAL, NUMERIC: знаковое число с плавающей запятой;

– DATE: дата;

– DATETIME: комбинация даты и времени;

– TIMESTAMP: отметка времени;

– TIME: время;

– YEAR: год в формате YY или YYYY;

– CHAR: строка фиксированного размера, дополняемая справа пробелами до максимальной длины;

– VARCHAR: строка переменной длины;

– TINYBLOB, TINYTEXT: двоичные или текстовые данные максимальной длиной 255 символов;

– BLOB, TEXT: двоичные или текстовые данные максимальной длиной 65535 символов;

– MEDIUMBLOB, MEDIUMTEXT: текст или двоичные данные;

– LONGBLOB, LONGTEXT: текст или двоичные максимальной данные длиной 4294967295 символов;

– ENUM: перечисление;

– SET: множества.

Поддерживаемые типы полей в Postgresql достаточно сильно отличаются, но позволяют записывать точно те же данные:

– bigint: знаковое 8-байтовое целое;

– bigserial: автоматически увеличиваемое 8-байтовое целое;

– bit: двоичная строка фиксированной длины;

– bit varying: двоичная строка переменной длины;

– boolean: флаг;

– box: прямоугольник на плоскости;

– byte: бинарные данные;

– character varying: строка символов фиксированной длины;

– character: строка символов переменной длины;

– cidr: сетевой адрес IPv4 или IPv6;

– circle: круг на плоскости;

– date: дата в календаре;

– double precision: число с плавающей запятой двойной точности;

– inet: адрес интернет IPv4 или IPv6;

– integer: знаковое 4-байтное целое число;
– interval: временной промежуток;
– line: бесконечная прямая на плоскости;
– lseg: отрезок на плоскости;
– macaddr: MAC-адрес;
– money: денежная величина;
– path: геометрический путь на плоскости;
– point: геометрическая точка на плоскости;
– polygon: многоугольник на плоскости;
– real: число с плавающей точкой одинарной точности;
– smallint: двухбайтовое целое число;
– serial: автоматически увеличиваемое четырехбитное целое число;
– text: строка символов переменной длины;
– time: время суток;
– timestamp: дата и время;
– tsquery: запрос текстового поиска;
– tsvector: документ текстового поиска;
– uuid: уникальный идентификатор;
– xml: XML-данные.

Как видно, типов данных в Postgresql больше и они более разнообразны, есть свои типы полей для определенных видов данных, которых нет MySQL. Отличие MySQL от Postgresql очевидно.

В качестве СУБД выбрана MySql, так как она обладает хорошим быстродействием, простотой настройкой, количество встроенных типов данных соответствует поставленной задаче автоматизации.

2.4 Построение схем функциональной структуры проектируемой системы

На рисунке 5 представлена обобщенная функциональная структура проектируемой АИС учета деятельности такси.

В рамках функциональной структуры также можно выделить такой набор модулей, которые выполняют вместе обобщенные функции:

– модуль взаимодействия с БД;

– модуль системной логики и обработки данных;

– модуль авторизации;

– модуль работы со справочниками;

– модуль выгрузки в Excel;

– модуль учета заявок на перевозку пассажиров;

– модуль инициализации и управления;

– модуль взаимодействия с пользователями.

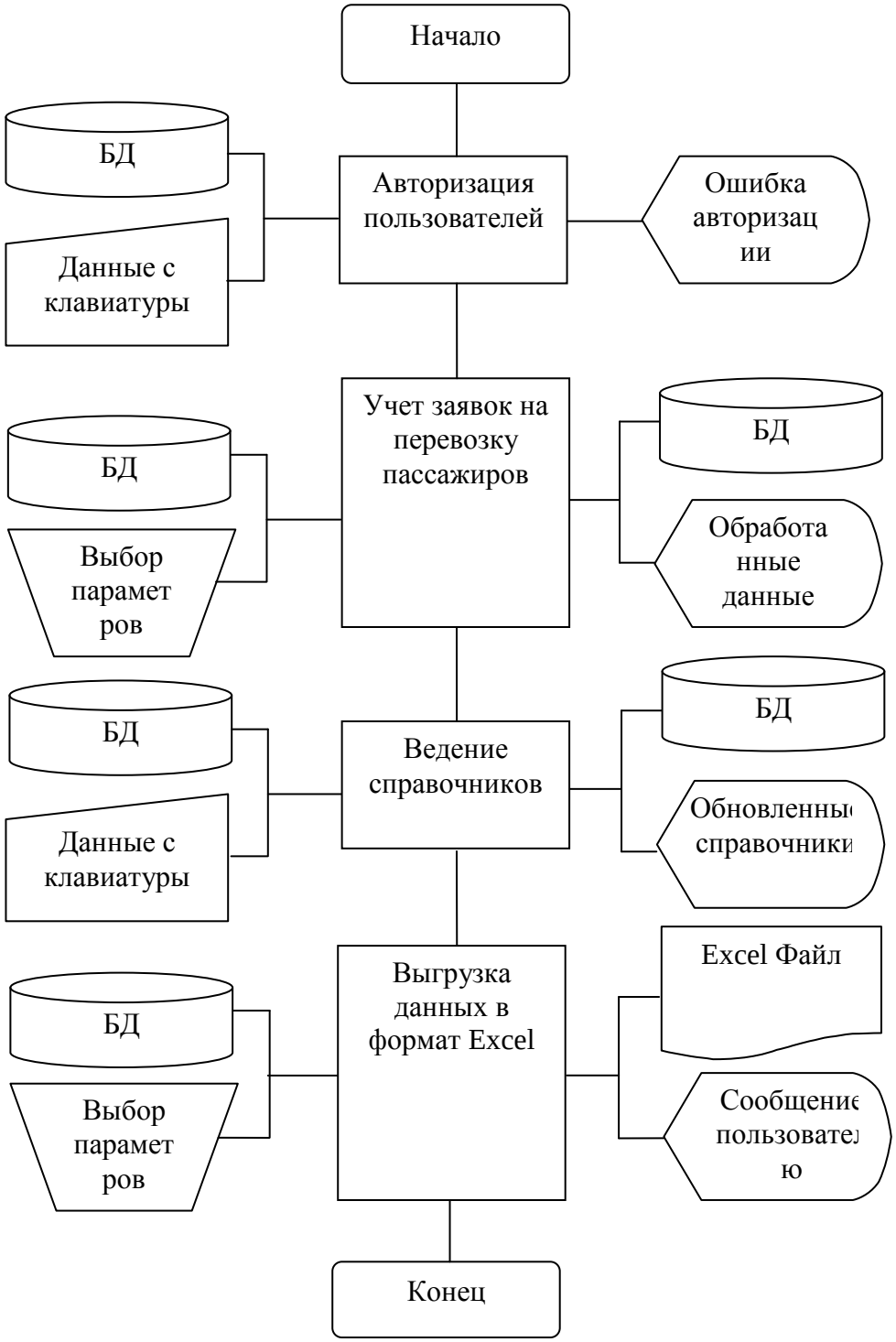


Рисунок 5 – Обобщенная функциональная структура проектируемой системы

Произведем детализацию функций системы путем раскрытия функций отдельных модулей. В таблице 4 представлены функции выделенных модулей.

Таблица 4 – Функции модулей

Название модуля	Функции
Модуль взаимодействия с БД	Отвечает за обеспечение связи приложения с БД. Отвечает за выполнение запросов к БД и предоставление для других модулей АИС унифицированных процедур для выполнения основных операций над данными: чтение, запись, изменение, удаление.
Модуль системной логики и обработки данных	Содержит в себе иерархию подчиняющихся форм приложения, что обеспечивает логичную и интуитивно понятную программную навигацию
Модуль авторизации	Обеспечивает проверку корректности учетных данных и информирование пользователя о результате проверки
Модуль работы со справочниками	Обеспечивает просмотр, редактирование, создание и удаление записей
Модуль выгрузки в Excel	Обеспечивает обработку таблицы на форме и запись данных в Excel файл
Модуль учета заявок на перевозку пассажиров	Обеспечивает просмотр, редактирование, создание и удаление записей
Модуль инициализации и управления	Содержит функции по первоначальной настройке ИС. Осуществляет также инициализацию вспомогательных подмодулей.
Модуль взаимодействия с пользователями	Обеспечивает работу стандартных форм пользовательского интерфейса, прием данных от пользователя и вывод результатов их обработки

Структура взаимодействия функциональных модулей представлена на рисунке 6.

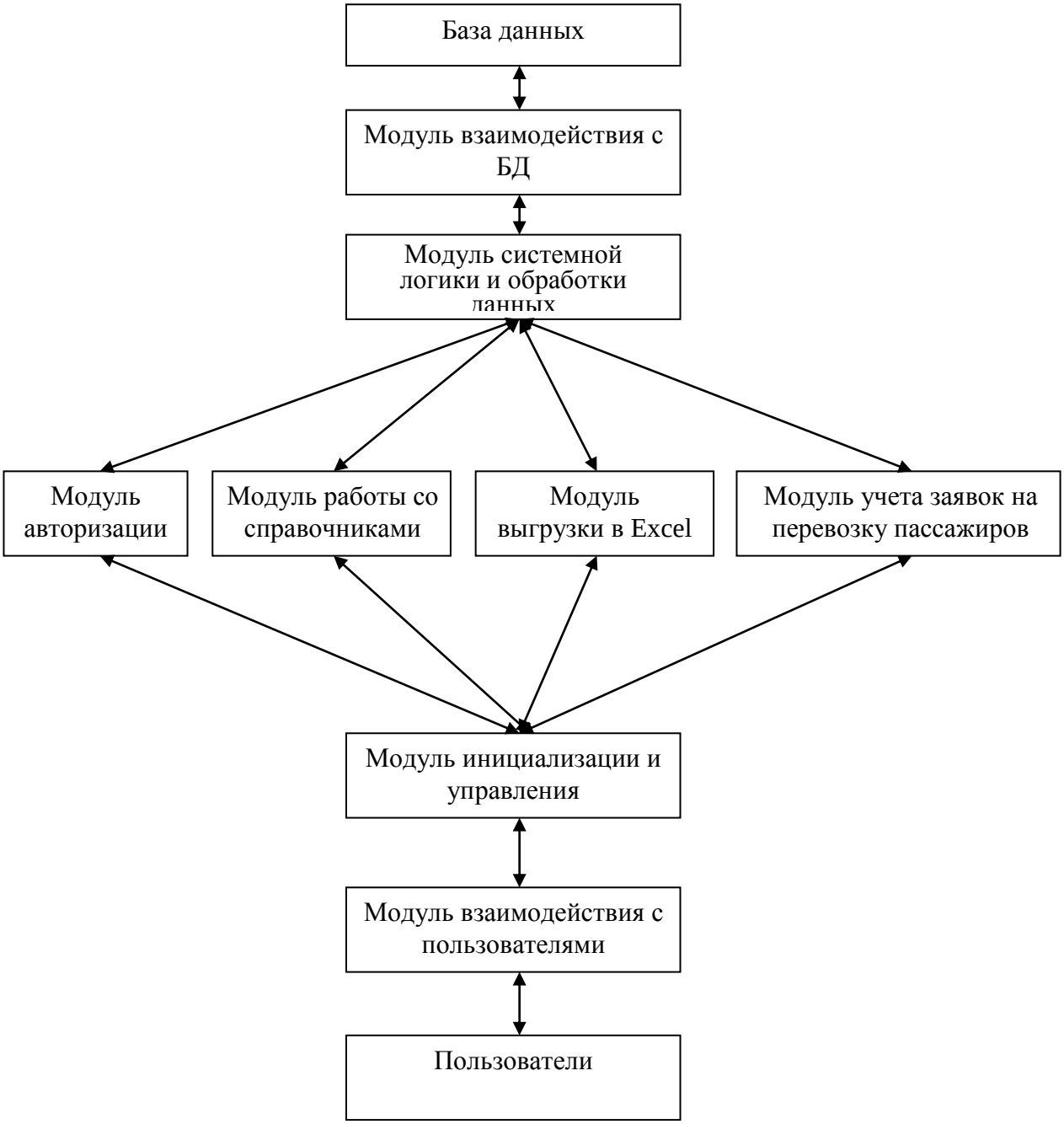


Рисунок 6 – Структура взаимодействия функциональных модулей

2.5 Проектирование базы данных

Для начала представим на рисунке 7 уровни моделей данных.

Рисунок 7 – Уровни моделей данных

Концептуальное (инфологическое) проектирование – построение семантической модели предметной области, то есть информационной модели наиболее высокого уровня абстракции. Такая модель создаётся без ориентации на какую-либо конкретную СУБД и модель данных.

Логическое (даталогическое) проектирование – создание схемы базы данных на основе конкретной модели данных, например, реляционной модели данных.

Физическое проектирование – создание схемы базы данных для конкретной СУБД.

Инфологическая модель

Инфологическая описание отображает реальный мир в некую совокупность концепций, полностью независимую от параметров среды хранения данных.

В результате инфологического проектирования можно выделить такие сущности:

- Работники;
- Клиенты;
- Тарифы;
- Доставка;
- Машины;
- Водители.

Была составлена инфологическая модель (рисунок 8).

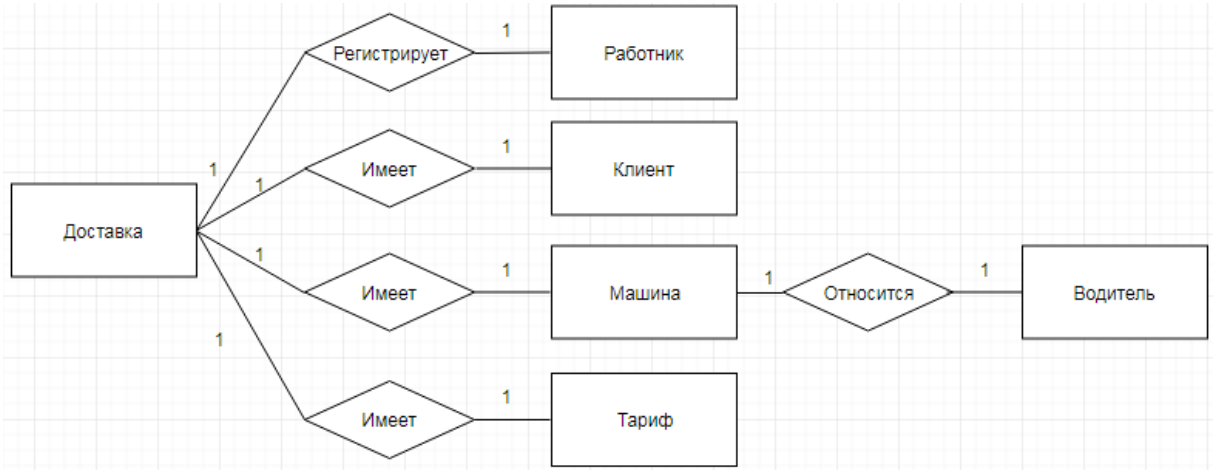


Рисунок 8 – Инфологическая модель

Сущность «Работники» обладает такими атрибутами:

- ФИО;
- Паспорт;
- Телефон;
- Дата поступления на работу;

- Логин;
- Пароль;
- Флаг администратора.

Сущность «Клиенты» обладает такими атрибутами:

- ФИО;
- Телефон;
- Почта;
- Дата регистрации.

Сущность «Тарифы» обладает такими атрибутами:

- Название;
- Информация;
- Цена за километр.

Сущность «Доставка» обладает такими атрибутами:

- Дата доставки;
- Километраж;
- Информация;
- Работник предприятия;
- Клиент;
- Тариф;
- Машина.

Сущность «Машины» обладает такими атрибутами:

- Гос. номер;
- Дата производства;
- Марка;
- Грузоподъемность;
- Специальное оборудование;
- Водитель.

Сущность «Водители» обладает такими атрибутами:

– ФИО;
– Водительские права;
– Адрес;
– Телефон;
– Паспорт;
– Номер договора.

Даталогическая модель

После построения инфологической модели данных перейдем к разработке даталогической и физической моделей данных.

В качестве модели данных будет использована реляционная. Эта модель подразумевает табличное хранение информации. Отметим, что эта модель исключает связь М-М, разбивая ее на две связи 1-М.

Также при построении даталогической модели данных учтем и принципы нормализации до третьей нормальной формы.

Первая нормальная форма – каждая запись в базе данных представляет один экземпляр сущности. Атомарность – поля не имеют дубликатов в каждой записи, и каждое поле содержит только одно значение.

Вторая нормальная форма – поля с не первичным ключом не должны быть зависимы от первичного ключа. Следование второй нормальной форме – это вопрос нахождения данных, которые часто дублируются в записях таблицы и которые могут принадлежать другой сущности.

Третья нормальная форма – не может быть транзитивных зависимостей между полями в таблице. Транзитивные зависимости между полями базы данных существует тогда, когда значения не ключевых полей зависят от значений других не ключевых полей.

Даталогическая модель представлена на рисунке 9.

Рисунок 9 – Даталогическая модель в ERWin

Так как выбрана СУБД MySql и она может некорректно работать с русскими названиями таблиц и атрибутов, то русские имена были заменены английскими (рисунок 10).

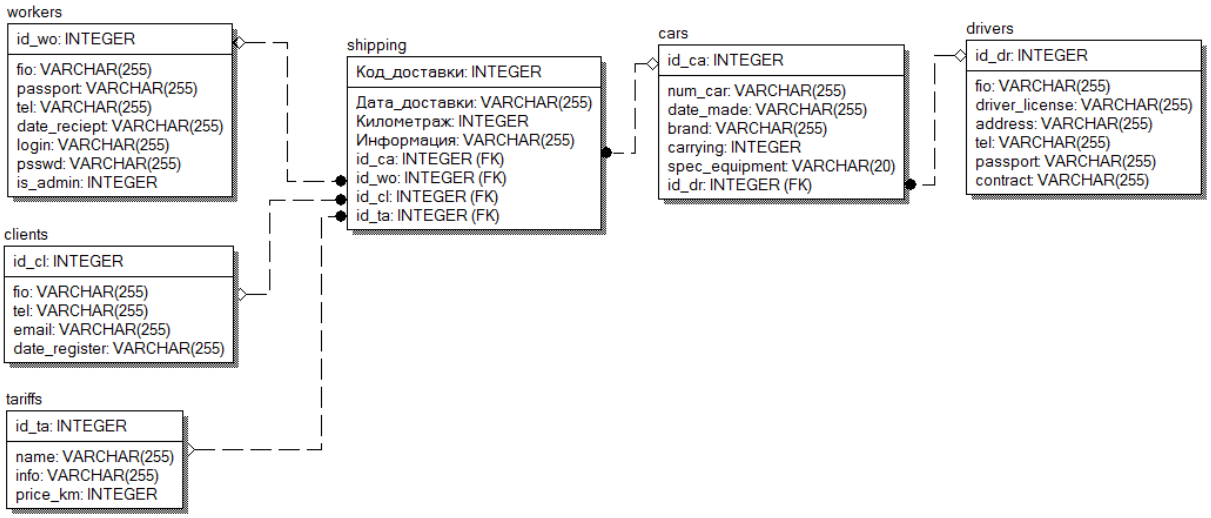


Рисунок 10 – Физическая модель в ERWin

Далее был получен скрипт создания базы данных, который был импортирован в клиент СУБД MySql – phpmyadmin. В его графическом дизайнере разработанная база данных выглядит так (рисунок 11).

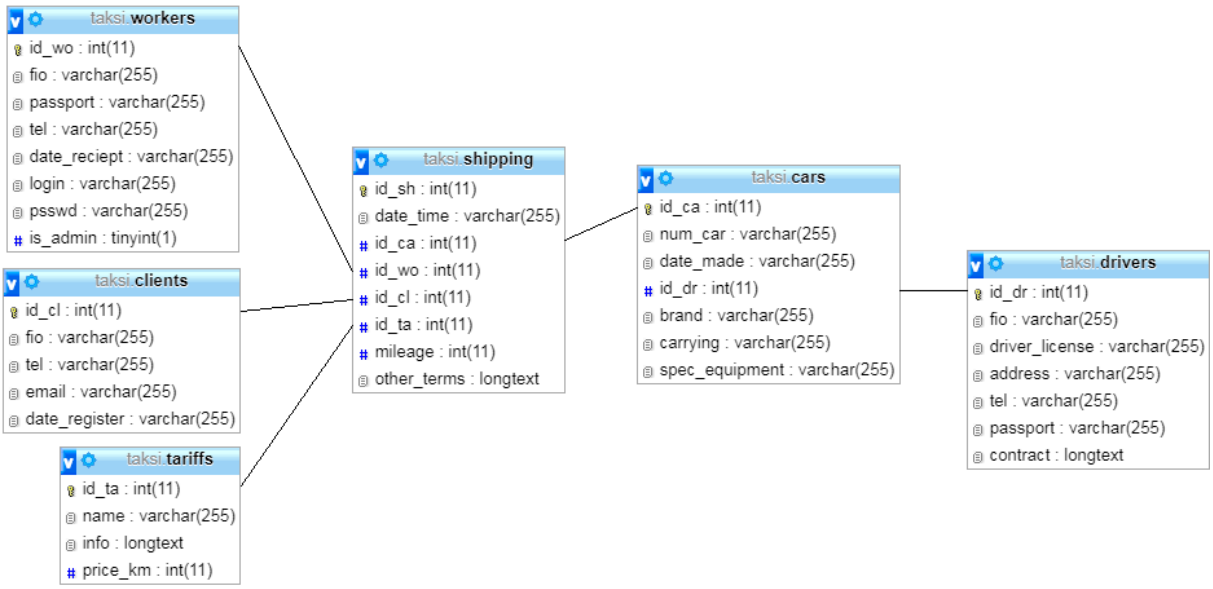


Рисунок 11 – База данных в СУБД MySql

Скрипт инициализации БД представлен в приложении Б.

2.6 Разработка интерфейса информационной системы

В широком смысле интерфейс представляет собой определенную стандартами границу между независимыми объектами, обменивающимися между собой некоторой информацией. Интерфейс задает параметры объектов, а также их процедуры и характеристики взаимодействия [12].

Интерфейс пользователя – это компоненты и элементы программы, способные оказывать непосредственное влияние на взаимодействие пользователя системы с самим программным обеспечением [10].

Поскольку в качестве программной среды для разработки выбрана Delphi, то представим макеты интерфейса, разработанные в рамках этой среды.

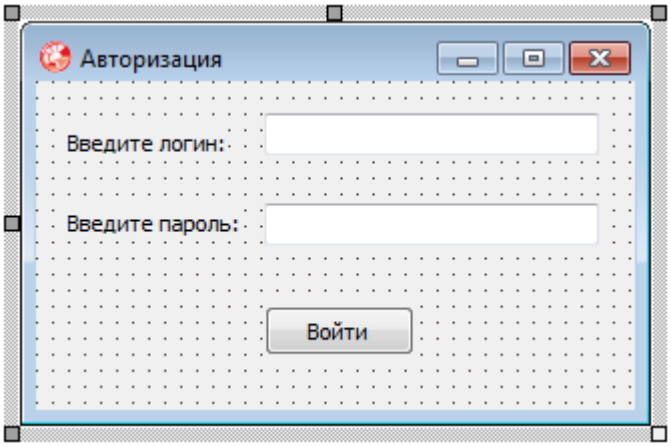


Рисунок 12 – Макет формы авторизации

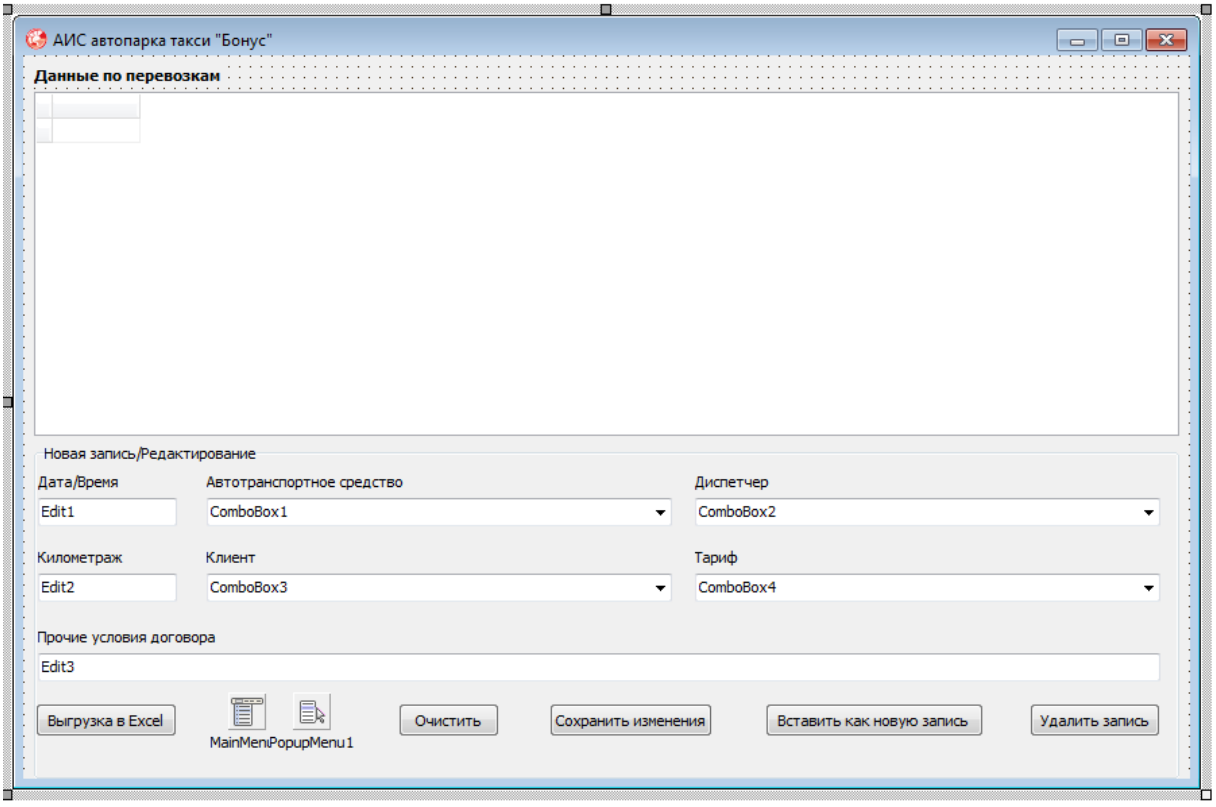


Рисунок 13 – Макет главной формы приложения

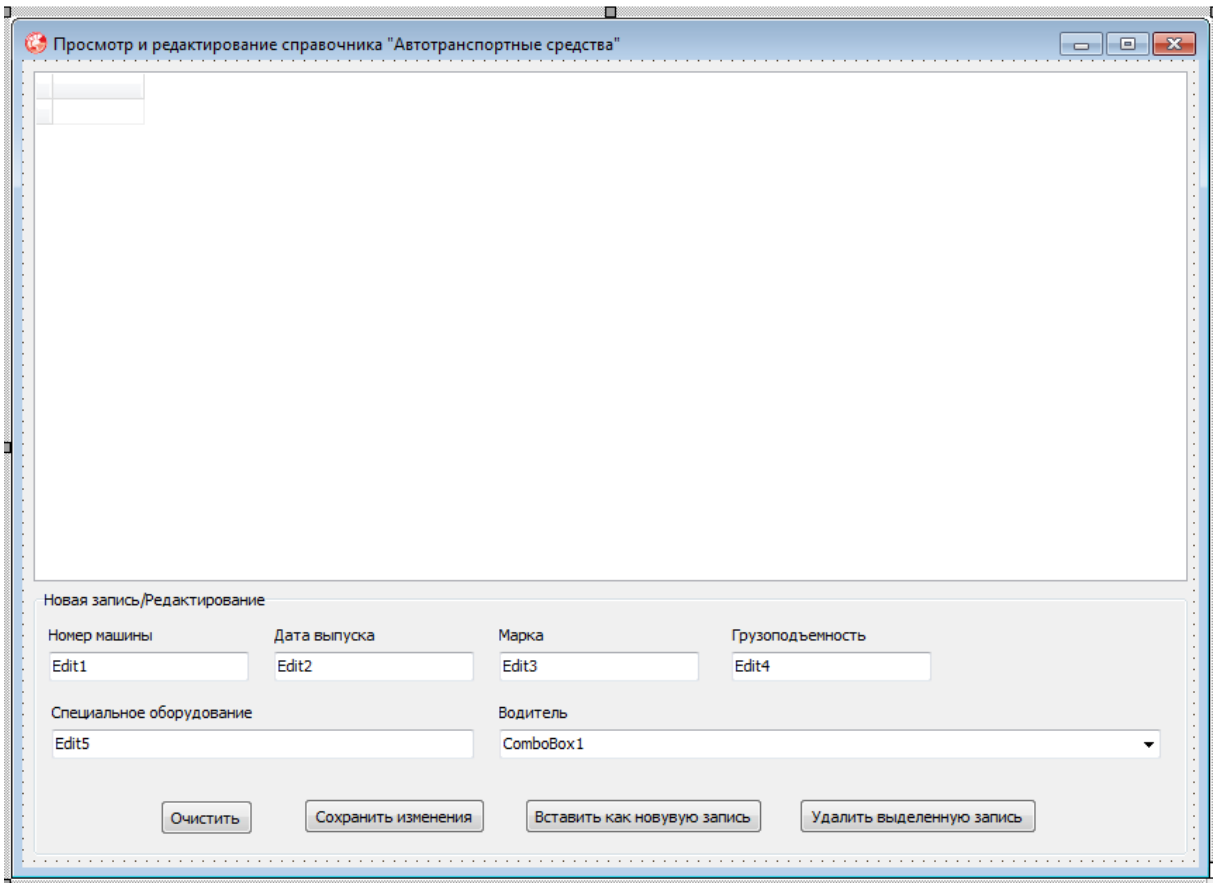


Рисунок 14 – Макет справочника «Автотранспортные средства»

2.7 Разработка алгоритма

При непосредственной реализации программного кода следует четко представлять последовательность действий программы. Это может быть достигнуто посредством разработки алгоритма.

Алгоритм представляет собой четкую последовательность действий, приводящих к определенному результату.

Рассмотрим алгоритм выгрузки сводных данных по перевозкам пассажиров в Excel файл.

Описание входной информации

На входе алгоритма:

– Программный модуль главной формы приложения;

Таблица с данными о перевозках клиентов.

Описание выходной информации

На выходе алгоритма записанный на носитель Excel файл с выгруженными данными

Описание алгоритма

Алгоритм выгрузки сводных данных по перевозкам представлен на рисунке 15. Алгоритм составлен с применение графического дизайнера MS Visio.

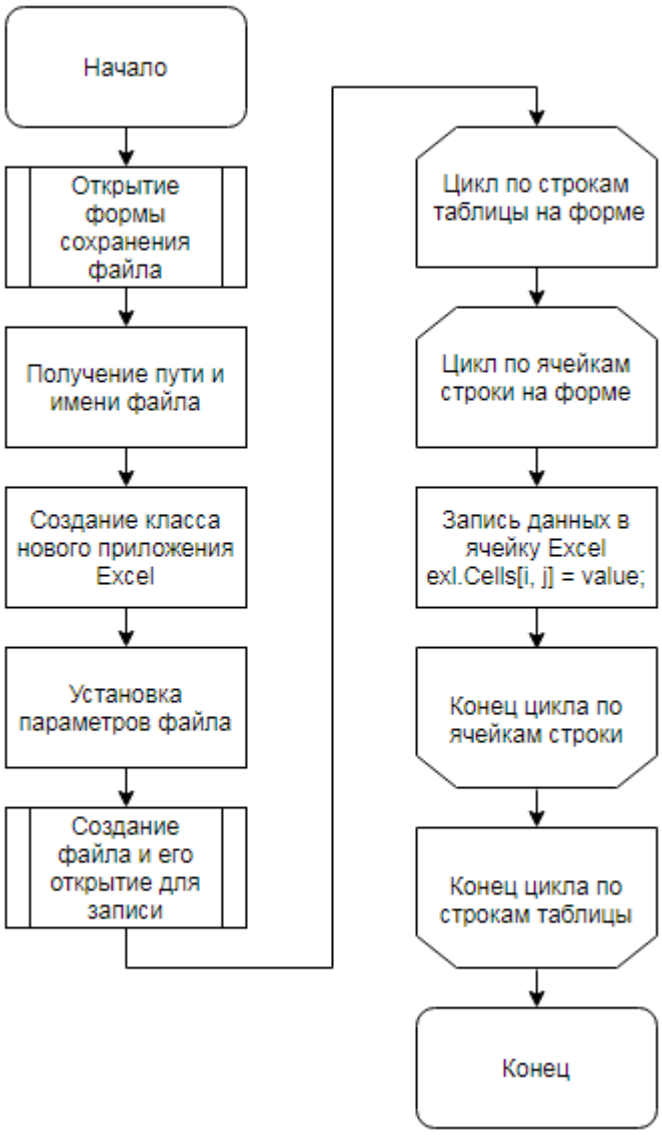


Рисунок 15 – Алгоритм выгрузки сводных данных по перевозкам

Первоначально происходит вызов интерфейса ОС Windows – формы сохранения файла, из которого АИС получает путь к файлу.

Затем происходит создание нового класса приложения Excel «Microsoft.Office.Interop.Excel.Application».

Потом идет установка параметров созданного приложения, после чего происходит непосредственное создание нового или открытие в режиме перезаписи существующего Excel файла.

В цикле по строкам выполняется цикл по ячейкам, копируя данные в excel файл.

2.8 Тестирование разработанного приложения

После запуска exe-файла загружается форма авторизации (рисунок 16).

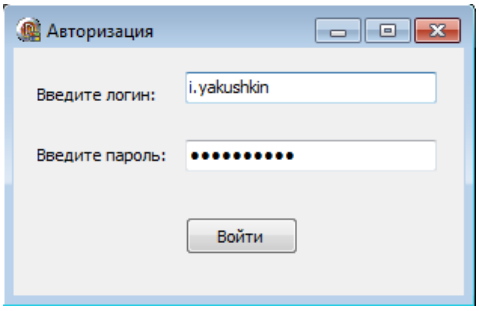


Рисунок 16 – Форма авторизации

После успешной авторизации загружается главная форма (рисунок 17), которая содержит таблицу с данными по перевозкам, а также управляющую панель снизу, которая позволяет вставлять новые записи, редактировать или удалять их.

Также имеется кнопка выгрузки данных в Excel, по нажатию на которую происходит открытие формы сохранения (рисунок 18).

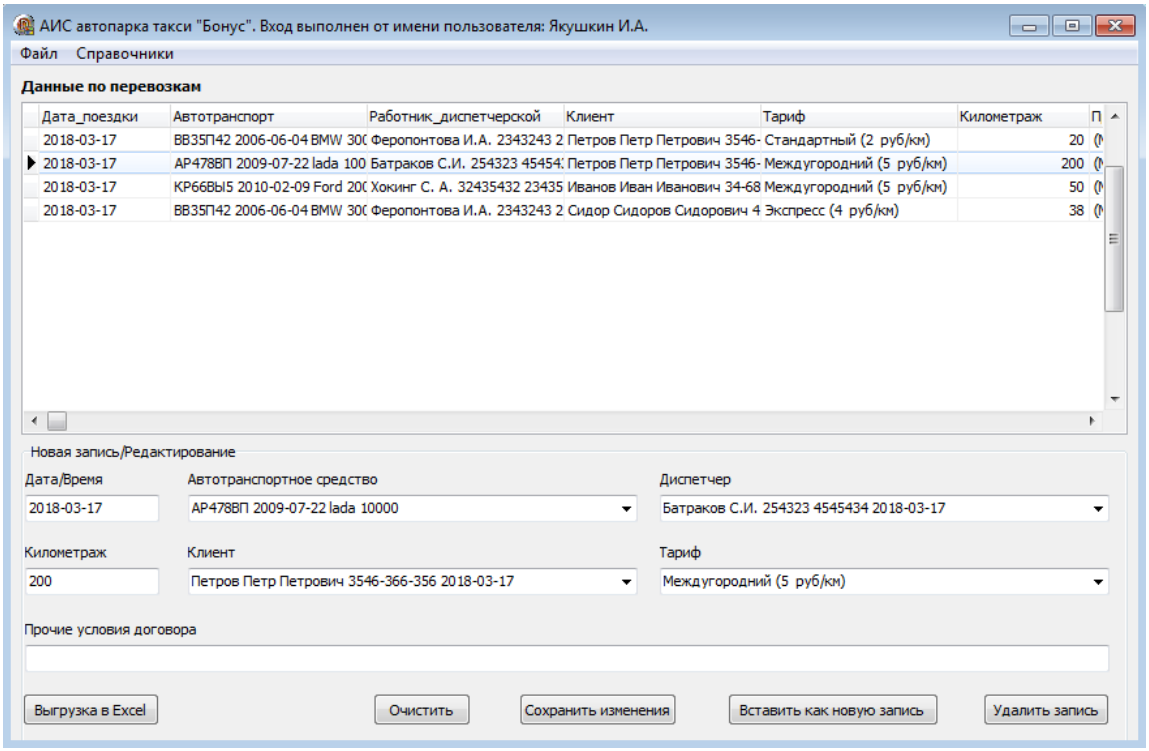


Рисунок 17 – Главная форма приложения

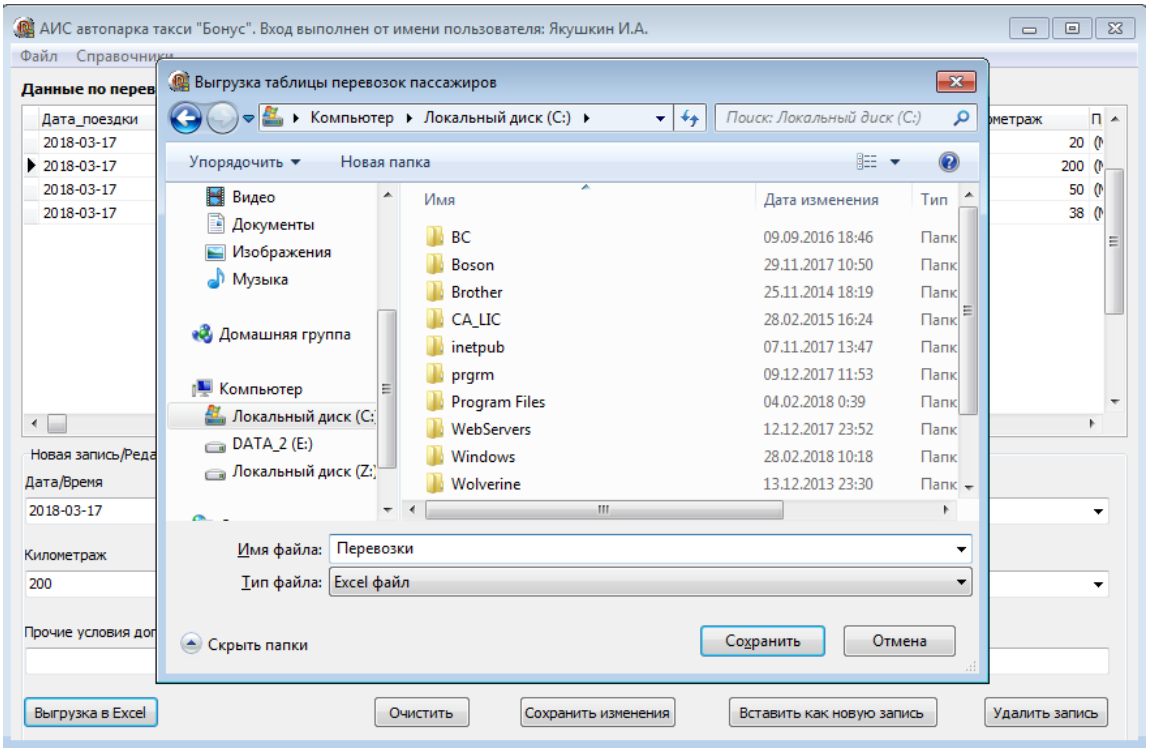
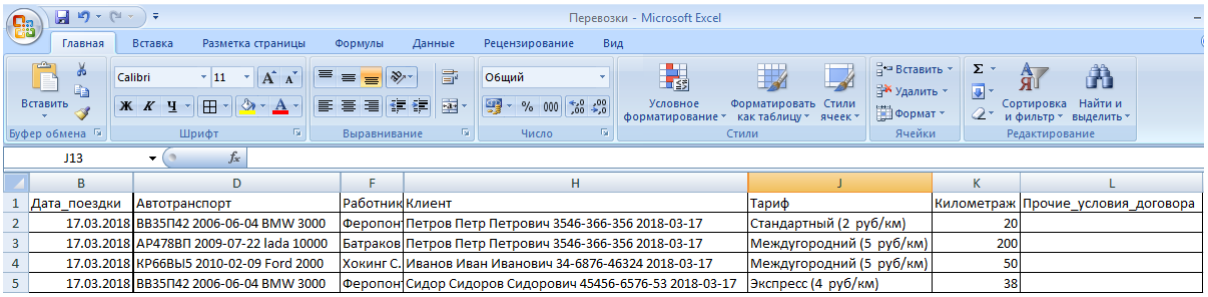


Рисунок 18 – Выгрузка данных в формат Excel

После сохранения файл можно просмотреть (рисунок 19). Это может быть полезно для бухгалтерии, например, чтобы сразу получить сводные данные по перевозкам.



	B	D	F	H	J	K	L
1	Дата_поездки	Автотранспорт	Работник	Клиент	Тариф	Километраж	Прочие_условия_договора
2	17.03.2018	ВВ35П42 2006-06-04 BMW 3000	Феропон	Петров Петр Петрович 3546-366-356 2018-03-17	Стандартный (2 руб/км)	20	
3	17.03.2018	АР478ВП 2009-07-22 lada 10000	Батраков	Петров Петр Петрович 3546-366-356 2018-03-17	Междугородний (5 руб/км)	200	
4	17.03.2018	КР66ВЫ5 2010-02-09 Ford 2000	Хокинг С.	Иванов Иван Иванович 34-6876-46324 2018-03-17	Междугородний (5 руб/км)	50	
5	17.03.2018	ВВ35П42 2006-06-04 BMW 3000	Феропон	Сидор Сидоров Сидорович 45456-6576-53 2018-03-17	Экспресс (4 руб/км)	38	

Рисунок 19 – Выгруженные данные

На рисунке 20 представлено меню справочников системы.

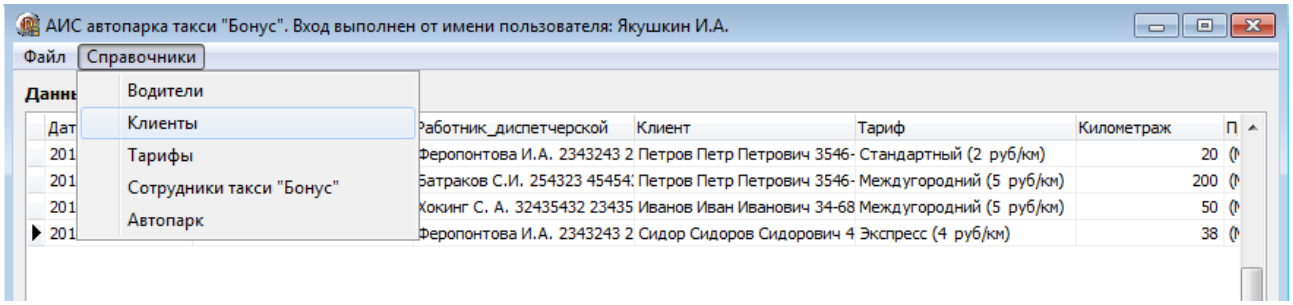


Рисунок 20 – Список справочников системы

Формат справочника машин представлен на рисунке 21. Данный справочник включает таблицу с данными по машинам, а также внизу имеется панель управления данными аналогично главной форме приложения.

На рисунке 22 представлен формат справочника водителей.

Форма обладает табличным данными, также внизу формы имеется специальный навигатор, который обладает интуитивно понятным интерфейсом и позволяет управлять записями. Назначения кнопок навигатора представлены в таблице 5.

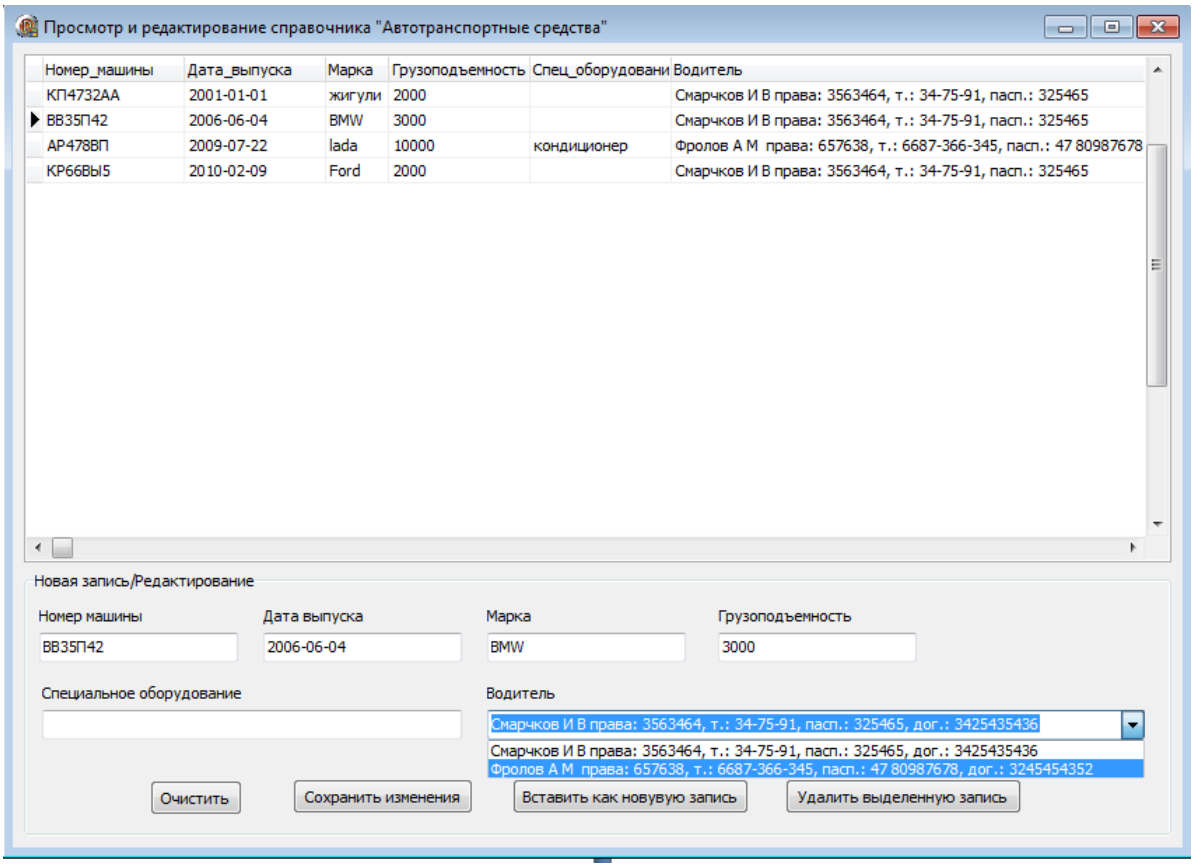


Рисунок 21 – Формат справочника машин

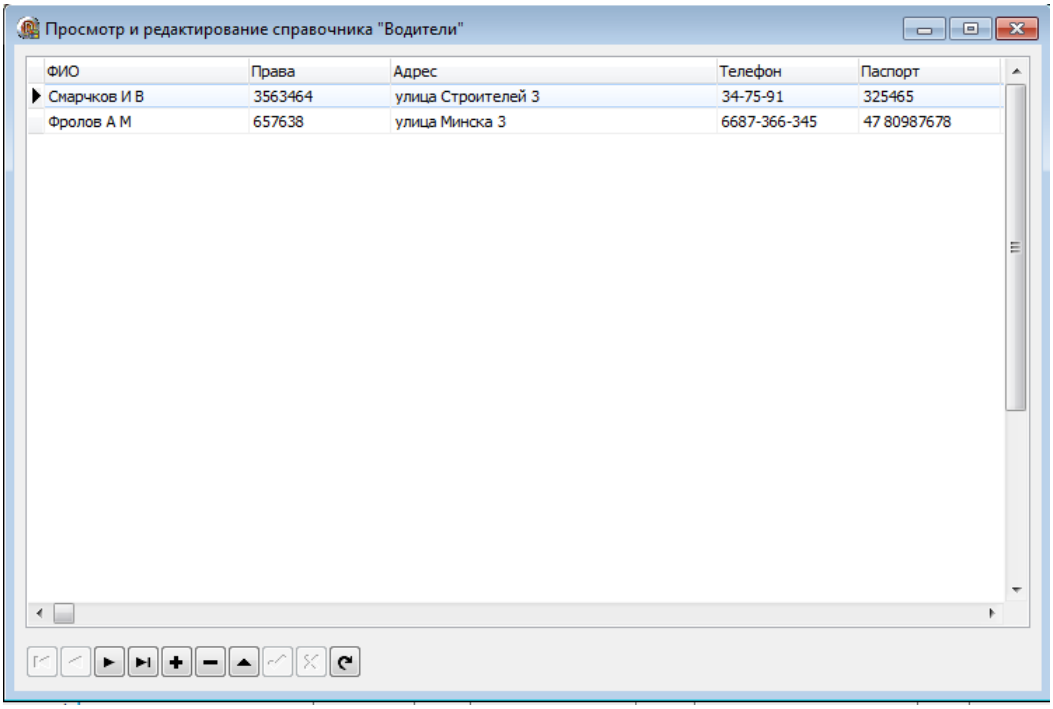


Рисунок 22 – Форма справочника водителей

Таблица 5 – Назначение кнопок навигатора

Кнопка	Краткое описание	Описание
	К первой	Указатель текущей записи перемещается к первой записи файла данных
	К предыдущей	Указатель текущей записи перемещается к предыдущей записи файла данных
	К следующей	Указатель текущей записи перемещается к следующей записи файла данных
	К последней	Указатель текущей записи перемещается к последней записи файла данных
	Добавить	В файл данных добавляется новая запись
	Удалить	Удаляется текущая запись файла данных
	Редактирование	Устанавливает режим редактирования текущей записи
	Сохранить	Изменения, внесенные в текущую запись, записываются в файл данных
	Отменить	Отменяет внесенные в текущую запись изменения
	Обновить	Обновляет записи из таблицы базы на форме

Остальные справочники системы имеют такой же формат, как и справочник водителей.

Таким образом, тестирование приложения показало его работоспособность и соответствие предъявляемым к нему требованиям.

Вывод: во второй главе были поставлены основные задачи на проектирование, составлено техническое задание на информационную систему.

В результате анализа различных языков программирования, сред разработки и СУБД, был выбран IDE Delphi, так как этот язык визуального программирования достаточно быстр в создании полноценного клиентского приложения, имеет расширенный список инструментальных средств разработки, и к тому же имеется легкий доступ к документации и поддержке со стороны производителя.

В качестве СУБД выбрана MySql, так как она обладает хорошим быстродействием, простотой настройки, количество встроенных типов данных соответствует поставленной задаче автоматизации. В ходе решения

поставленных задач на проектирование и программирование была разработана АИС «Автопарк такси Бонус» и успешно протестирована.

ЗАКЛЮЧЕНИЕ

В рамках данной работы выполнено проектирование информационной системы для ООО «Такси Бонус».

«Такси Бонус» – это активно развивающаяся молодая организация, предоставляющая услуги такси в городе Лесосибирск. Компания была основана в феврале 2014 года.

В качестве средства анализа и проектирования выбраны BPWin, а также ERWin.

Выполнено моделирование бизнес процессов предприятия в BPWin. Недостатком существующего учета деятельности такси является то, что применяется бумажный журнал учета, которые не позволяет вести оперативный учет и доступ к информации, серьезно затрудняет и замедляет ее анализ.

Поэтому принято решение рекомендовать предприятию перевести учет в электронный вид, причем организовать это стоит при помощи информационной системы, чтобы в дальнейшем была возможность расширять функционал по мере роста компании.

В качестве способа достижения данной задачи решено выполнить самостоятельную разработку информационной системы.

Согласно ГОСТ 34.602.89 [3] составлена постановка задачи на проектирование.

В качестве языка программирования выбран Delphi, а в качестве СУБД – MySQL.

Выполнена поэтапная разработка базы данных, которая включила в себя инфологическое описание, инфологическую модель, даталогическую и физическую модели. Составлен скрипт инициализации БД.

Также составлена функциональная схема приложения, выявлены и описаны ее модули, разработаны макеты форм пользовательского интерфейса.

Тестирование приложения показало его работоспособность и соответствие предъявляемым к нему требованиям.

В итоге решены такие задачи:

– рассмотрена деятельность предприятия в области учета автопарка такси и его работы;

– обоснована потребность в автоматизации;

– выполнен обзор и выбор инструментальных средств проектирования и реализации системы;

– выполнена поэтапная разработка и тестирование автоматизированной информационной системы.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Агапов, В. Профессиональная разработка программного обеспечения / В. Агапов. – Москва : Символ-Плюс, 2013. – 240 с.

2. Волкова, В. Теория систем и системный анализ. Учебник / В. Волкова. – Москва : Юрайт, 2016. – 464 с.

3. ГОСТ 34.602.89 Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы. – Взамен ГОСТ 24.201-85; введ. 01.01.1990. – Москва : Стандартинформ, 2009. – 9 с.

4. Дубейковский, В. И. Практика функционального моделирования с AllFusion Process Modeler 4.1. Где? Зачем? Как? / В. И. Дубейковский. – Москва : «ДИАЛОГ-МИФИ», 2014. – 464 с.

5. Калянов, Г. Н. Моделирование, анализ, реорганизация и автоматизация бизнес-процессов / Г. Н. Калянов. – Москва : Финансы и статистика, 2016. – 267 с.

6. Качала, В. Теория систем и системный анализ / В. Качала. – Москва : Academia, 2013. – 272 с.

7. Козленко, Л. Проектирование информационных систем / Л. Козленко // КомпьютерПресс. – № 9. – 2013. – С. 3-17.

8. Куприянов, Д. Информационное обеспечение профессиональной деятельности. Учебник и практикум для СПО / Д. Куприянов. – Москва : Юрайт, 2017. – 255 с.

9. Купер, А. Интерфейс. Основы проектирования и взаимодействия / А. Купер. – Санкт-Петербург : Питер, 2018. – 720 с.

10. Маклаков, С. В. BPwin и ERwin. CASE – средства разработки информационных систем / С. В. Маклаков. – Москва : «ДИАЛОГ- МИФИ», 2013. – 256 с.

11. Маклафлин, Б. PHP и MySQL. Исчерпывающее руководство / Б. Маклафин. – Санкт-Петербург : Питер, 2016. – 544 с.

12. Мандел, Т. Дизайн интерфейсов / Т. Мандел. – Москва : ДМК, 2015. – 410 с.

13. Репин, В. В. Процессный подход к управлению. Моделирование бизнес-процессов / В. В. Репин, В. Г. Елиферов. – Москва : Манн, Иванов и Фербер, 2013. – 408 с.

14. Советов, Б. Я. Архитектура информационных систем: учебник для студ. учреждений высш. проф. образования / Б. Я. Советов. – Москва : Издательский центр «Академия», 2012. – 288 с.

15. Анализ организационной структуры предприятия [Электронный ресурс]. – Режим доступа: http://www.fox-manager.ru/analiz-organizacionnoj-struktury-predpriyatiya.html

16. Типы организационных структур управления [Электронный ресурс]. – Режим доступа: http://pagelooker.org/rezyume/2-uncategorised/16-tipy-organizatsionnykh-struktur-upravleniya-chast-1.html

17. Сравнение интегрированных сред разработки [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/Сравнение_интегрированных_сред_разработки

18. СУБД PostgreSQL [Электронный ресурс]. – Режим доступа: htps://ru.wikipedia.org/wiki/PostgreSQL

19. ERwinDataModeler [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/ERwin_Data_Modeler

20. Top 10 Programming Languages [Электронный ресурс]. – Режим доступа: http://spectrum.ieee.org/computing/software/top-10-programming-languages

ПРИЛОЖЕНИЕ А

Свидетельство, удостоверяющее факт публикации

МОЛОДОЙ УЧЁНЫЙ

СВИДЕТЕЛЬСТВО

о публикации научной статьи

Настоящим подтверждается, что

Якушкин
Иван Александрович

студент

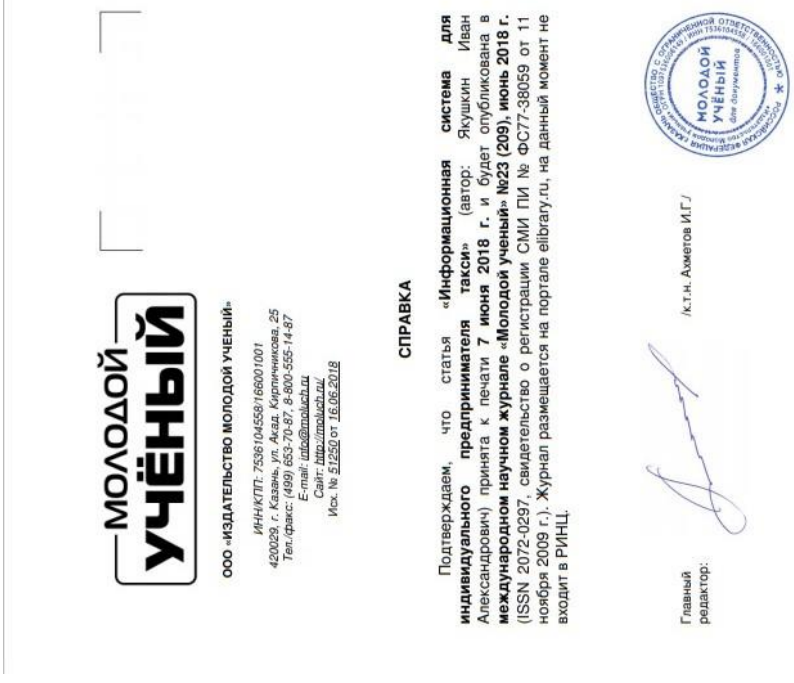
Лесосибирский педагогический институт – филиал Сибирского федерального университета
является автором статьи, опубликованной в международном научном журнале «Молодой ученый» (№23 (209), июнь 2018 г.)

«Информационная система для индивидуального предпринимателя такси»

https://moluch.ru/archive/209/51250/

Главный редактор
Издательства «Молодой ученый»
к.т.н. Ахметов И.Г.

СМ № 0209-0098Z



МОЛОДОЙ УЧЁНЫЙ

ООО «ИЗДАТЕЛЬСТВО МОЛОДОЙ УЧЕНЫЙ»

ИНН/КПП: 7536104558/166001001
420029, г. Казань, ул. Акад. Кирпичникова, 25
Тел./факс: (499) 653-70-87, 8-800-555-14-87
E-mail: info@moluch.ru
Сайт: http://moluch.ru/
Исх. № 51250 от 16.06.2018

СПРАВКА

Подтверждаем, что статья **«Информационная система для индивидуального предпринимателя такси»** (автор: Якушкин Иван Александрович) принята к печати **7 июня 2018 г.** и будет опубликована в **международном научном журнале «Молодой ученый» №23 (209), июнь 2018 г.** (ISSN 2072-0297, свидетельство о регистрации СМИ ПИ № ФС77-38059 от 11 ноября 2009 г.). Журнал размещается на портале elibrary.ru, на данный момент не входит в РИНЦ.

Главный редактор: /К.Т.Н. Ахметов И.Г./

Исп.: Осянина Е.И.

ПРИЛОЖЕНИЕ Б

Скрипт инициализации БД

```
-- phpMyAdmin SQL Dump
-- version 3.5.1
-- http://www.phpmyadmin.net
--
-- Хост: 127.0.0.1
-- Время создания: Мар 17 2018 г., 13:48
-- Версия сервера: 5.5.25
-- Версия PHP: 5.3.13

SET SQL_MODE="NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";


/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;

--
-- База данных: `taksi`
--

-- --------------------------------------------------------

--
-- Структура таблицы `cars`
--

CREATE TABLE IF NOT EXISTS `cars` (
  `id_ca` int(11) NOT NULL AUTO_INCREMENT,
  `num_car` varchar(255) DEFAULT NULL,
  `date_made` varchar(255) DEFAULT NULL,
  `id_dr` int(11) DEFAULT NULL,
  `brand` varchar(255) DEFAULT NULL,
  `carrying` varchar(255) DEFAULT NULL,
  `spec_equipment` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`id_ca`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 AUTO_INCREMENT=5 ;

--
-- Дамп данных таблицы `cars`
--

INSERT INTO `cars` (`id_ca`, `num_car`, `date_made`, `id_dr`, `brand`,
`carrying`, `spec_equipment`) VALUES
(1, 'КП4732АА', '2001-01-01', 1, 'жигули', '2000', NULL),
(2, 'ВВ35П42', '2006-06-04', 1, 'BMW', '3000', NULL),
(3, 'АР478ВП', '2009-07-22', 2, 'lada', '10000', 'кондиционер'),
(4, 'КР66ВЫ5', '2010-02-09', 1, 'Ford', '2000', NULL);

-- --------------------------------------------------------

--
-- Структура таблицы `clients`
--
```

```
CREATE TABLE IF NOT EXISTS `clients` (
  `id_cl` int(11) NOT NULL AUTO_INCREMENT,
  `fio` varchar(255) DEFAULT NULL,
  `tel` varchar(255) DEFAULT NULL,
  `email` varchar(255) DEFAULT NULL,
  `date_register` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`id_cl`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 AUTO_INCREMENT=4 ;

--
-- Дамп данных таблицы `clients`
--

INSERT INTO `clients` (`id_cl`, `fio`, `tel`, `email`, `date_register`) VALUES
(1, 'Иванов Иван Иванович', '34-6876-46324', 'ivan@gmail.com', '2018-03-17'),
(2, 'Петров Петр Петрович', '3546-366-356', 'petr@mail.ru', '2018-03-17'),
(3, 'Сидор Сидоров Сидорович', '45456-6576-53', 'sr@yandex.ru', '2018-03-17');

-- --------------------------------------------------------

--
-- Структура таблицы `drivers`
--

CREATE TABLE IF NOT EXISTS `drivers` (
  `id_dr` int(11) NOT NULL AUTO_INCREMENT,
  `fio` varchar(255) DEFAULT NULL,
  `driver_license` varchar(255) DEFAULT NULL,
  `address` varchar(255) DEFAULT NULL,
  `tel` varchar(255) DEFAULT NULL,
  `passport` varchar(255) DEFAULT NULL,
  `contract` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`id_dr`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 AUTO_INCREMENT=3 ;

--
-- Дамп данных таблицы `drivers`
--

INSERT INTO `drivers` (`id_dr`, `fio`, `driver_license`, `address`, `tel`,
`passport`, `contract`) VALUES
(1, 'Смарчков И В', '3563464', 'улица Строителей 3', '34-75-91', '325465',
'3425435436'),
(2, 'Фролов А М ', '657638', 'улица Минска 3', '6687-366-345', '47 80987678',
'3245454352');

-- --------------------------------------------------------

--
-- Структура таблицы `shipping`
--

CREATE TABLE IF NOT EXISTS `shipping` (
  `id_sh` int(11) NOT NULL AUTO_INCREMENT,
  `date_time` varchar(255) DEFAULT NULL,
  `id_ca` int(11) DEFAULT NULL,
  `id_wo` int(11) DEFAULT NULL,
  `id_cl` int(11) DEFAULT NULL,
  `id_ta` int(11) DEFAULT NULL,
  `mileage` int(11) DEFAULT NULL,
  `other_terms` longtext,
```

```
  PRIMARY KEY (`id_sh`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 AUTO_INCREMENT=5 ;

--
-- Дамп данных таблицы `shipping`
--

INSERT INTO `shipping` (`id_sh`, `date_time`, `id_ca`, `id_wo`, `id_cl`,
`id_ta`, `mileage`, `other_terms`) VALUES
(1, '2018-03-17', 2, 4, 2, 1, 20, ''),
(2, '2018-03-17', 3, 2, 2, 2, 200, ''),
(3, '2018-03-17', 4, 3, 1, 2, 50, ''),
(4, '2018-03-17', 2, 4, 3, 3, 38, '');

-- --------------------------------------------------------

--
-- Структура таблицы `tariffs`
--

CREATE TABLE IF NOT EXISTS `tariffs` (
  `id_ta` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(255) DEFAULT NULL,
  `info` varchar(255) DEFAULT NULL,
  `price_km` int(11) DEFAULT NULL,
  PRIMARY KEY (`id_ta`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 AUTO_INCREMENT=4 ;

--
-- Дамп данных таблицы `tariffs`
--

INSERT INTO `tariffs` (`id_ta`, `name`, `info`, `price_km`) VALUES
(1, 'Стандартный', 'для обычных перевозок', 2),
(2, 'Междугородний', NULL, 5),
(3, 'Экспресс', NULL, 4);

-- --------------------------------------------------------

--
-- Структура таблицы `workers`
--

CREATE TABLE IF NOT EXISTS `workers` (
  `id_wo` int(11) NOT NULL AUTO_INCREMENT,
  `fio` varchar(255) DEFAULT NULL,
  `passport` varchar(255) DEFAULT NULL,
  `tel` varchar(255) DEFAULT NULL,
  `date_reciept` varchar(255) DEFAULT NULL,
  `login` varchar(255) NOT NULL,
  `psswd` varchar(255) NOT NULL,
  `is_admin` tinyint(1) NOT NULL DEFAULT '0',
  PRIMARY KEY (`id_wo`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 AUTO_INCREMENT=5 ;

--
-- Дамп данных таблицы `workers`
--

INSERT INTO `workers` (`id_wo`, `fio`, `passport`, `tel`, `date_reciept`,
`login`, `psswd`, `is_admin`) VALUES
```

```
(1, 'Грызлов В.В.', '65565', '234342', '2018-03-17', 'gryzlov', 'g^a#E2_h95',
1),
(2, 'Батраков С.И.', '254323', '4545434', '2018-03-17', 'batrakov',
'0Zv.8Vg#e9', 0),
(3, 'Хокинг С. А.', '32435432', '2343543', '2018-03-17', 'hoking', 'f%0_V%7a!',
0),
(4, 'Феропонтова И.А.', '2343243', '23434532', '2018-03-17', 'feropontova',
'd)2d_^N1q1', 0);

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
```

ПРИЛОЖЕНИЕ В

Листинги программных модулей

Главная форма приложения

```
unit Main;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, DB, ADODB, Grids, DBGrids, StdCtrls, MSExcel, ExtCtrls, DBCtrls,
  Generics.Collections,
  Auth, Catalog, Cars,
  Menus, ToolWin, ActnMan, ActnCtrls, ActnMenus, ComCtrls;

type
  MyList = TList <TStringList>;
  MyHash = TDictionary<string, string>;
  MyDict = TDictionary<Integer, Integer>;

  TForm1 = class(TForm)
    MainMenu1: TMainMenu;
    Label1: TLabel;
    PopupMenu1: TPopupMenu;
    DBGrid1: TDBGrid;
    GroupBox1: TGroupBox;
    Edit1: TEdit;
    Edit2: TEdit;
    Edit3: TEdit;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    ComboBox1: TComboBox;
    ComboBox2: TComboBox;
    Label8: TLabel;
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    Button4: TButton;
    Label9: TLabel;
    Button5: TButton;
    Label5: TLabel;
    Label6: TLabel;
    ComboBox3: TComboBox;
    ComboBox4: TComboBox;
    procedure FormCreate(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure DBGrid1CellClick(Column: TColumn);
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure Button5Click(Sender: TObject);
  private
    { Private declarations }
    procedure catalogItemClick(Sender: TObject);
    procedure carsItemClick(Sender: TObject);
    procedure exitItemClick(Sender: TObject);
```

```
  public
    { Public declarations }
    ADOConnection1: TADOConnection;
    sqlResult: MyList;
    id_wo, fio: string;
    logged, is_admin: Boolean;

    catalogs: MyHash;
    titles: MyHash;
    ids: MyHash;
    widths: MyHash;

    procedure execSQL(sql:string);
    procedure openCatalog(name: string);
    procedure showSQLinGridView(sql: string; grid: TStringGrid; headers: string;
widths: string);
    procedure showSQLinComboBox(sql: string; var dic: MyDict; combo: TComboBox);
    procedure clearGridView(grid: TStringGrid);
    function  getIndInComboByValue(v: Integer; var dic: MyDict): Integer;
    procedure loadCars;
    procedure loadShipping;
    procedure clearData;
  end;

const
  // Параметры подключения к СУБД
  SERVER:   String = '127.0.0.1';
  DATABASE: String = 'taksi';
  USER:     String = 'root';
  PSSWRD:   String = '';

  IDWIDTH:  String = '0';
  HALLO:    String = 'АИС автопарка такси "Бонус". Вход выполнен от имени
пользователя: ';

var
  Form1: TForm1;
  wo, cl, ta, ca: MyDict;

implementation

{$R *.dfm}

procedure TForm1.FormShow(Sender: TObject);
var
  key: string;
  itm, itm1: TMenuItem;
begin
  // Вывод окна авторизации
  Form1.logged := False;
  Form2.ShowModal;
  if Form1.logged = False then
    Form1.Close;

  itm := TMenuItem.Create(nil);
  itm.Caption := 'Файл';
  itm1 := TMenuItem.Create(nil);
  itm1.Caption := 'Выход';
  itm1.OnClick := exitItemClick;
  itm.Add(itm1);
  MainMenu1.Items.Add(itm);
```

```
  // Формирование списка справочников
  catalogs := MyHash.Create;
  if Form1.is_admin = True then
    catalogs.Add('Сотрудники такси "Бонус"', 'workers');
  catalogs.Add('Клиенты', 'clients');
  catalogs.Add('Тарифы', 'tariffs');
  catalogs.Add('Водители', 'drivers');

  titles := MyHash.Create;
  if Form1.is_admin = True then
    titles.Add('Сотрудники такси "Бонус"',
'ID;ФИО;Паспорт;Телефон;Дата_приема;Логин;Пароль;Администратор?');
  titles.Add('Клиенты', 'ID;ФИО;Телефон;Email;Дата_регистрации');
  titles.Add('Тарифы', 'ID;Наименование;Описание;Цена_за_км');
  titles.Add('Водители', 'ID;ФИО;Права;Адрес;Телефон;Паспорт;Договор');

  ids := MyHash.Create;
  if Form1.is_admin = True then
    ids.Add('Сотрудники такси "Бонус"',
'id_wo;fio;passport;tel;date_reciept;login;psswd;is_admin');
  ids.Add('Клиенты', 'id_cl;fio;tel;email;date_register');
  ids.Add('Тарифы', 'id_ta;name;info;price_km');
  ids.Add('Водители', 'id_dr;fio;driver_license;address;tel;passport;contract');

  widths := MyHash.Create;
  if Form1.is_admin = True then
    widths.Add('Сотрудники такси "Бонус"', '35;200;70;70;90;70;70;100');
  widths.Add('Клиенты', '35;200;100;100;100');
  widths.Add('Тарифы', '35;200;200;100');
  widths.Add('Водители', '35;200;100;200;100;100;100');

  itm := TMenuItem.Create(nil);
  itm.Caption := 'Справочники';
  for key in catalogs.Keys do begin
    itm1 := TMenuItem.Create(itm);
    itm1.Caption := key;
    itm1.OnClick := catalogItemClick;
    itm.Add(itm1);
  end;
  itm1 := TMenuItem.Create(itm);
  itm1.Caption := 'Автопарк';
  itm1.OnClick := carsItemClick;
  itm.Add(itm1);
  MainMenu1.Items.Add(itm);

  // Загрузка полей
  Form1.showSQLinComboBox('select id_ca, concat(num_car, '' '', date_made, ''
'', brand, '' '', carrying) as car from cars', ca, Form1.ComboBox1);
  Form1.showSQLinComboBox('select id_wo, concat(fio, '' '', passport, '' '',
tel, '' '', date_reciept) as worker from workers', wo, Form1.ComboBox2);
  Form1.showSQLinComboBox('select id_cl, concat(fio, '' '', tel, '' '',
date_register) as client from clients', cl, Form1.ComboBox3);
  Form1.showSQLinComboBox('select id_ta, concat(name, '' ('', price_km, ''
руб/км)'') as tarif from tariffs', ta, Form1.ComboBox4);
  Form1.Edit1.Clear;
  Form1.Edit2.Clear;
  Form1.Edit3.Clear;
  loadShipping;
end;
```

```
procedure TForm1.DBGrid1CellClick(Column: TColumn);
begin
  Form1.Edit1.Text := Form1.DBGrid1.DataSource.DataSet.Fields[1].AsString;
  Form1.Edit2.Text := Form1.DBGrid1.DataSource.DataSet.Fields[10].AsString;
  Form1.Edit3.Text := Form1.DBGrid1.DataSource.DataSet.Fields[11].AsString;
  Form1.ComboBox1.ItemIndex :=
Form1.getIndInComboByValue(Form1.DBGrid1.DataSource.DataSet.Fields[2].AsInteger,
ca);
  Form1.ComboBox2.ItemIndex :=
Form1.getIndInComboByValue(Form1.DBGrid1.DataSource.DataSet.Fields[4].AsInteger,
wo);
  Form1.ComboBox3.ItemIndex :=
Form1.getIndInComboByValue(Form1.DBGrid1.DataSource.DataSet.Fields[6].AsInteger,
cl);
  Form1.ComboBox4.ItemIndex :=
Form1.getIndInComboByValue(Form1.DBGrid1.DataSource.DataSet.Fields[8].AsInteger,
ta);

//ID;Дата_поездки;ID;Автотранспорт;ID;Работник_диспетчерской;ID;Клиент;ID;Тариф;
Километраж;Прочие_условия_договора
end;

procedure TForm1.exitItemClick(Sender: TObject);
begin
  Form1.Close;
end;

procedure TForm1.Button1Click(Sender: TObject); // Очистить
begin
  Form1.clearData;
end;

procedure TForm1.Button2Click(Sender: TObject); // Сохранить
var
  sql: string;
  id_sh, date_time, id_ca, id_wo, id_cl, id_ta, mileage, other_terms: string;
begin

//ID;Дата_поездки;ID;Автотранспорт;ID;Работник_диспетчерской;ID;Клиент;ID;Тариф;
Километраж;Прочие_условия_договора
  if(Form1.DBGrid1.DataSource.DataSet.Fields.Count > 0) then begin
    id_sh       := Form1.DBGrid1.DataSource.DataSet.Fields[0].AsString;
    date_time   := Form1.Edit1.Text;
    mileage     := Form1.Edit2.Text;
    other_terms := Form1.Edit3.Text;
    id_ca       := IntToStr(ca[Form1.ComboBox1.ItemIndex]);
    id_wo       := IntToStr(wo[Form1.ComboBox2.ItemIndex]);
    id_cl       := IntToStr(cl[Form1.ComboBox3.ItemIndex]);
    id_ta       := IntToStr(ta[Form1.ComboBox4.ItemIndex]);

    sql := 'update shipping set date_time='''+date_time+''',
mileage='''+mileage+''', other_terms='''+other_terms+''', id_ca='''+id_ca+''',
id_wo='''+id_wo+''', id_cl='''+id_cl+''', id_ta='''+id_ta+''' where
id_sh='+id_sh;
    Form1.execSQL(sql);
    Form1.loadShipping;
    Form1.clearData;
  end;
end;

procedure TForm1.Button3Click(Sender: TObject); // Новая
```

```
var
  sql: string;
  id_sh, date_time, id_ca, id_wo, id_cl, id_ta, mileage, other_terms: string;
begin
  if(Form1.DBGrid1.DataSource.DataSet.Fields.Count > 0) then begin
    id_sh       := Form1.DBGrid1.DataSource.DataSet.Fields[0].AsString;
    date_time   := Form1.Edit1.Text;
    mileage     := Form1.Edit2.Text;
    other_terms := Form1.Edit3.Text;
    id_ca       := IntToStr(ca[Form1.ComboBox1.ItemIndex]);
    id_wo       := IntToStr(wo[Form1.ComboBox2.ItemIndex]);
    id_cl       := IntToStr(cl[Form1.ComboBox3.ItemIndex]);
    id_ta       := IntToStr(ta[Form1.ComboBox4.ItemIndex]);

    sql := 'insert into shipping(id_sh, date_time, id_ca, id_wo, id_cl, id_ta,
mileage, other_terms) values('''+date_time+''', '''+mileage+''',
'''+other_terms+''', '+id_ca+', '+id_wo+', '+id_cl+', '+id_ta+',
'+Form1.id_wo+')';
    Form1.execSQL(sql);
    Form1.loadShipping;
    Form1.clearData;
  end;
end;

procedure TForm1.Button4Click(Sender: TObject); // Удалить
var
  sql,id_sh: string;
begin
  if(Form1.DBGrid1.DataSource.DataSet.Fields.Count > 0) then begin
    if Application.MessageBox(PChar('Вы уверены что хотите удалить эту
запись?'), 'Внимание!!!', MB_OKCANCEL)=id_OK then begin
      id_sh := Form1.DBGrid1.DataSource.DataSet.Fields[0].AsString;
      sql := 'delete from shipping where id_sh='+id_sh;
      Form1.execSQL(sql);
      Form1.loadShipping;
      Form1.clearData;
    end;
  end;
end;

procedure TForm1.Button5Click(Sender: TObject); // Выгрузка в Excel
  var
  saveDialog : TSaveDialog; // Переменная диалога сохранения
begin
  // Создание объекта диалога сохранения - назначая его нашей переменной диалога
сохранения
  saveDialog := TSaveDialog.Create(self);

  // Give the dialog a title
  saveDialog.Title := 'Выгрузка таблицы перевозок пассажиров';

  // Установка начального каталога
  saveDialog.InitialDir := GetCurrentDir;

  // Разрешаем сохранять файлы типа .txt и .doc
  saveDialog.Filter := 'Excel файл|*.xls|Excel файд|*.xls';

  // Установка расширения по умолчанию
  saveDialog.DefaultExt := 'xls';

  // Выбор текстовых файлов как стартовый тип фильтра
```

```
  saveDialog.FilterIndex := 1;

  // Отображение диалог сохранения файла
  if saveDialog.Execute then begin
    export2xls(saveDialog.FileName, DBGrid1);
    ShowMessage('Данные выгружены');
  end;

  // Освобождения диалога
  saveDialog.Free;
end;

procedure TForm1.catalogItemClick(Sender: TObject);
var
  item: TMenuItem;
begin
  item := TMenuItem(Sender);
  openCatalog(item.Caption);
end;

procedure TForm1.carsItemClick(Sender: TObject);
begin
  loadCars;
  Form4.ShowModal();
end;

procedure TForm1.loadCars;
// Открыть справочник автотранспортных средств
var
  ind, i: Integer;
  sql: string;
  strs, wdths: TStringList;
  ADOQuery1: TADOQuery;
  DataSource1: TDataSource;
begin
  name := Trim(name);
  if name = '' then Exit;

  ADOQuery1 := TADOQuery.Create(nil);
  DataSource1 := TDataSource.Create(nil);

  Form4.DBGrid1.DataSource := DataSource1;

  ADOQuery1.Active := False;
  sql := 'select a.id_ca, a.num_car, a.date_made, a.brand, a.carrying,
a.spec_equipment, b.id_dr, concat(b.fio, '' права: '', b.driver_license, '', т.:
'', b.tel, '', пасп.: '', b.passport) as driver '+
  'from cars as a ' +
  'inner join drivers as b on a.id_dr=b.id_dr';

  ADOQuery1.SQL.Add(sql);
  ADOQuery1.Connection := Form1.ADOConnection1;
  DataSource1.DataSet := ADOQuery1;
  DataSource1.AutoEdit := True;
  ADOQuery1.Open();
  DataSource1.Enabled := True;

  strs := TStringList.Create;
  strs.Delimiter := ';';
  strs.StrictDelimiter := False;
```

```
  strs.DelimitedText :=
'ID;Номер_машины;Дата_выпуска;Марка;Грузоподъемность;Спец_оборудование;ID;Водите
ль';
  wdths := TStringList.Create;
  wdths.Delimiter := ';';
  wdths.StrictDelimiter := False;
  wdths.DelimitedText := '35;100;100;70;100;100;35;350';
  for i := 0 to Form4.DBGrid1.Columns.Count - 1 do begin
    Form4.DBGrid1.Columns[i].Title.caption := strs.Strings[i];
    Form4.DBGrid1.Columns[i].Width := StrToInt(wdths.Strings[i]);
  end;

  Form4.Caption := 'Просмотр и редактирование справочника "Автотранспортные
средства"';
  Form4.DBGrid1.Columns[0].Visible := false;
  Form4.DBGrid1.Columns[6].Visible  := false;
end;

procedure TForm1.loadShipping;
var
  ind, i: Integer;
  sql: String;
  strs, wdths: TStringList;
  ADOQuery1: TADOQuery;
  DataSource1: TDataSource;
begin
  name := Trim(name);
  if name = '' then Exit;

  ADOQuery1 := TADOQuery.Create(nil);
  DataSource1 := TDataSource.Create(nil);

  Form1.DBGrid1.DataSource := DataSource1;

  ADOQuery1.Active := False;
  sql := 'select a.id_sh, a.date_time, b.id_ca, concat(b.num_car, '' '',
b.date_made, '' '', b.brand, '' '', b.carrying) as car, c.id_wo, concat(c.fio,
'' '', c.passport, '' '', c.tel, '' '', c.date_reciept) as worker, ' +
  'd.id_cl, concat(d.fio, '' '', d.tel, '' '', d.date_register) as client,
e.id_ta, concat(e.name, '' ('', e.price_km, ''  руб/км)'') as tarif, a.mileage,
a.other_terms ' +
  'from shipping as a ' +
  'inner join cars as b on a.id_ca=b.id_ca ' +
  'inner join workers as c on a.id_wo=c.id_wo ' +
  'inner join clients as d on a.id_cl=d.id_cl ' +
  'inner join tariffs as e on a.id_ta=e.id_ta ' ;

  ADOQuery1.SQL.Add(sql);
  ADOQuery1.Connection := Form1.ADOConnection1;
  DataSource1.DataSet := ADOQuery1;
  DataSource1.AutoEdit := True;
  ADOQuery1.Open();
  DataSource1.Enabled := True;

  strs := TStringList.Create;
  strs.Delimiter := ';';
  strs.StrictDelimiter := False;
  strs.DelimitedText :=
'ID;Дата_поездки;ID;Автотранспорт;ID;Работник_диспетчерской;ID;Клиент;ID;Тариф;К
илометраж;Прочие_условия_договора';
  wdths := TStringList.Create;
```

```
  wdths.Delimiter := ';';
  wdths.StrictDelimiter := False;
  wdths.DelimitedText := '35;100;35;150;35;150;35;150;35;150;100;150';
  for i := 0 to Form1.DBGrid1.Columns.Count - 1 do begin
    Form1.DBGrid1.Columns[i].Title.caption := strs.Strings[i];
    Form1.DBGrid1.Columns[i].Width := StrToInt(wdths.Strings[i]);
  end;

  Form1.DBGrid1.Columns[0].Visible := false;
  Form1.DBGrid1.Columns[2].Visible  := false;
  Form1.DBGrid1.Columns[4].Visible  := false;
  Form1.DBGrid1.Columns[6].Visible  := false;
  Form1.DBGrid1.Columns[8].Visible := false;
end;

procedure TForm1.clearData;
begin
  Form1.Edit1.Clear;
  Form1.Edit2.Clear;
  Form1.Edit3.Clear;
  Form1.ComboBox1.ItemIndex := -1;
  Form1.ComboBox2.ItemIndex := -1;
  Form1.ComboBox3.ItemIndex := -1;
  Form1.ComboBox4.ItemIndex := -1;
end;

end.
```